Frontier Assignment for Multi-Agent Exploration Under Communication Constraints

1st Marshall Vielmetti
Department of Robotics
University of Michigan
Ann Arbor, USA
mvielmet@umich.edu

***Abstract*—Multi-robot exploration is a fundamental problem in robotics, with many applications in search-and-rescue, environmental monitoring, and more. In order for multi-robot exploration to be the most effective, robots must be able to communicate with each other, and coordinate their efforts. This paper explores the use of a frontier-based exploration algorithm in the context of a multi-robot system, and combining this with a communication-aware task-planning strategy to ensure connectivity is maintained throughout the exploration process.**

***Index Terms*—Multi-Agent Exploration, Global Connectivity, Frontier-Based Exploration**

I. Introduction

Multi-Robot Systems (MRS) and their applications to robot exploration have been the target of extensive recent research, particularly due to their ability of such systems to cover vastly more area than a single robot, in a fraction of the time.

This paper explores two key challenges with extending conventional exploration algorithms to be applicable to MRS. First, the robots must be able coordinate their exploration efforts, which is necessary to prevent wasted effort.

Secondly, and in order to accomplish the first, the robots must be able to communicate. There are many distinct approaches to maintaining this communication link, as well as varying requirements.

This paper will explore the use of a frontier-based exploration algorithm in the context of a multi-robot system, and combining this with a communication-aware task-planning strategy to ensure connectivity is maintained throughout the exploration process.

A. Multi-Robot Exploration

Robotic exploration is one of the long-standing fundamental problems in robotics, and much recent work has been done to extend this to multi-agent systems. Methods for single robot exploration include frontier-based algorithms yamauchiFrontierbasedApproachAutonomous1997, or potential-fields amigoniMultirobotExplorationCommunicationRestricted2017 which both work to drive agents towards unexplored areas. However, when introducing a multi-agent framework, choosing which agent is assigned to which frontier, how to select which frontiers to explore given the position of agents, or how to construct the problem in a distributed setting become quickly non-trivial.

Works such as burgardCoordinatedMultirobotExploration2005, andreCoordinatedMultirobotExploration2014 seek to extend these algortihms by introducing a centralized planner. Other works assign roles to agents based on their relative positions within the environment dehoogRoleBasedAutonomousMultirobot2009.

Many of these recent works focus specifically on teams which are able to communicate, and thus improve the efficiency of their exploration efforts. However, in many settings, this communication is non-trivial. For example, in a search-and-rescue scenario, one of the common examples for robotic exploration, centralized communication infrastructure such as cell towers may not be available. In these cases, robots must be able to maintain inter-agent connectivity over ad-hoc networks qiuHeterogeneousAdHoc2017,falconiGraphBasedAlgorithmRobotic2012.

This paper will be mostly interested in a centralized approach to exploration, in which a global server or leader performs task assignment to all agents, which then execute those tasks.

B. Communication Constraints

There are many sub-categories of communication constraints which can be imposed on a multi-robot system. Communication can be modeled as circular, line of site, or even as a signal, which is impacted by obstacles in the environment amigoniMultirobotExplorationCommunicationRestricted2017. There are also different connecvitity requirements, such as global connectivity, where all robots must be able to communicate with all other robots, or local connectivity, where robots must only be able to communicate with a subset of other robots. This can also be broken up as event-based connectivity, where robots must be able to communicate with other robots at specific times, or continuous connectivity, where robots must be able to communicate at all times.

Furthermore, some papers consider a base-station, with which robots must relay information or otherwise maintain connectivity with throughout their exploration mission. This has applications such as requiring off-board computing power located at a basestation, access to global information, or even needing to stream or relay information back to human operators mukhijaTwoPhaseRecursive2010.

This paper will focus on maintaining continuous, global connectivity among all agents in the system, without a base station. This paper will also assume a circular communication model to simplify the problem.

II. Problem Formulation

This paper makes a handful of simplifying assumptions in order to focus on the core problem of exploration under communication constraints.

Most significantly, the paper will assume perfect sensors and localization. This is a common assumption in the literature with regards to this topic caregnatonetoRealtimeMotionPlanning2022, and eliminates the multi-robot SLAM problem, which is a significant challenge in its own right and outside the scope of this paper.

The paper will also assume a circular communication mode, which ignores intereference from obstacles or other robots. This is a simplifying assumption, but is common in the literature amigoniMultirobotExplorationCommunicationRestricted2017.

Furthermore, the paper assumes no communication bandwidth, and that as long as global connectivity is maintained, it is possible for a centralized, arbitrary 'leader' to plan and assign tasks to all agents in the network. Given that the goal of the paper is to design such an algorithm that maintains this global connectivity, this is a reasnoable, albeit significant, assumption.

This paper will also consider only simple, single-integrator dynamics, with input constraints, again as a simplifying assumption.

A. Approach

This paper seeks to apply lyapunov-like barrier functions to the objective of maintaining global connectivity. This will allow agents to act locally with respect to their motion planning, without worrying about violating the connectivity guarantees, panagouDistributedCoordinationControl2016a

Furthermore, the paper will implement a market-based algorithm to allow agents to request the removal of barriers, as long as global connectivity is maintained. This approach was first proposed in michaelMaintainingConnectivityMobile2009 to maintain connectivity even when multiple simulatneous deletion requests are made.

This will be reflected by dynamically adding and removing the recentered barrier functions, which will allow agents to move freely within the environment, as long as they maintain their required connectivity.

While some previous works have attempted to impose a static structure on the connectivity graph, this paper allows the graph to be dynamic, and additionally the application of the R-LBF is believed to be novel in this context.

B. Mathematical Formulation

We consider a multi-robot system with n agents, with $x_i(t) \in \mathbb{R}^2$ representing the position of agent i at time t. Each agent is assumed to have a communication radius r_c and sensing radius r_s. It is assumed $r_c > r_s$.

We consider a time-varying, undirected graph $\mathcal{G}(t) = (\mathcal{V}, \mathcal{E}_e(t))$, where $\mathcal{V} = \{1, 2, \dots, n\}$ is a set of vertices corresponding to each agent. The edgeset, $\mathcal{E}_e(t)$, is defined as a time varying set, with the elemnets representing an enforced communication link between agents, which is itself a subset of all possible edges that could exist given the communication radius. That is:

$$\mathcal{E}_e(t) \subseteq \mathcal{E}(t) = \{(i,j) \mid ||x_i - x_j|| \leq r_c,\ i,j \in \mathcal{V}\} \tag{1}$$

Global connectivity is defined as the property that there exists a path between any two agents in the network. We can check connectivity of the graph by checking the second smallest eigenvalue of the Laplacian matrix of the graph, $\mathcal{L}(t)$, which is defined as:

$$\mathcal{L}(t) = \mathcal{D}(t) - \mathcal{A}(t) \tag{2}$$

Where $\mathcal{D}(t)$ is the degree matrix, and $\mathcal{A}(t)$ is the adjacency matrix of the graph at time t, michaelMaintainingConnectivityMobile2009.

For each element of the adjacency matrix, we define a Lyapunov-Like Barrier function, $B_{ij}(x_i, x_j)$, which is a function of the relative position of agents i and j. This function is defined as:

$$B_{ij}(x_i, x_j) = \frac{1}{2}||x_i - x_j||^2 - r_c^2 \tag{3}$$

C. Feasability Considerations

The feasability of the problem is a significant concern. In particular, consider the case where two agent's actions are in conflict. In this case, the agents will be unable to move, and the system will be unable to continue exploring.

To address this, we introduce two different roles. The first is the 'explorer' role, which is responsible for expanding the frontier. The second is the 'relay' role, which is responsible for maintaining connectivity. We initialize all agents as explorers, and then dynamically assign the relay role to agents which are in conflict, based on their expected information gain. This approach has been considered before, for example in dehoogRoleBasedAutonomousMultirobot2009.

Agents assigned the relay role will no longer seek to explore, and will instead maneuvre to maintain connectivity.

D. Mapping and Localization

As stated in the introduction, this paper makes a number of assumptions, key among which is perfect sensing, localization, and information sharing.

This paper uses an occupancy grid map (OGM) approach to represent the environment. At each iteration of the algorithm, every cell within each agent's sensing radius (accounting for obstacles obstructing LOS) is updated as either a 'free' or 'occupied' cell.

This map is then used by the high-level task planning algorithm to determine areas of the map which have not yet been explored.

Additionally, the map is used to determine the cost of moving an agent to a given location, as agents will only be routed through areas which are 'free' in the map.

E. Task-Planning Algorithm

The high-level task planning algorithm seeks to assign each agent a target location, which maximizes the information gain of the system as a whole, while maintaining the connectivity requirements.

1) For each agent, calculate the cost of moving to each cell in the map using Djikstra's algorithm.
2) Calculate the information gain of each cell in the map, using a frontier-based exploration algorithm.
3) Find a global optimal assignment of agents to cells, which maximizes the information gain of the system as a whole while minimizing the cost of moving agents to their assigned cells, and maintaining connectivity.

Let:

- I_{ij} be the information gain of assigning agent i to cell j.
- C_{ij} be the cost of moving agent i to cell j.
- a_{ij} be a binary variable representing whether agent i is assigned to cell j.
- n be the number of agents.
- m be the number of cells in the map.
- r_c be the communication radius of the agents.
- $\mathcal{E}_e(t)$ be the enforced communication links at time t.
- y_j be the position of cell j.

This optimization problem is formalized as:

$$\begin{aligned} &\max \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}(I_{ij} - C_{ij}) \\ &\text{subject to:} \\ &\sum_{i=1}^{n} a_{ij} \leq 1, \forall j, \\ &\sum_{j=1}^{m} a_{ij} = 1, \forall i, \\ &a_{ij}a_{kl}||y_j - y_l||_2 \leq r_c, \forall (i,k) \in \mathcal{E}_e(t), \forall j, l \end{aligned} \tag{4}$$

Where the first constraint limits the number of agents assigned to a cell, the second constraint ensures each agent is assignet to exactly one cell, and the third constraint ensures that connected agents are assigned to cells within their communication radius. Unfortunately this problem was beyond the capabilities of my comptuer to solve even for relatively small problems.

III. Task Planning Simulations

We will begin by simulating the task planning algorithm in a simple environment, with three agents.

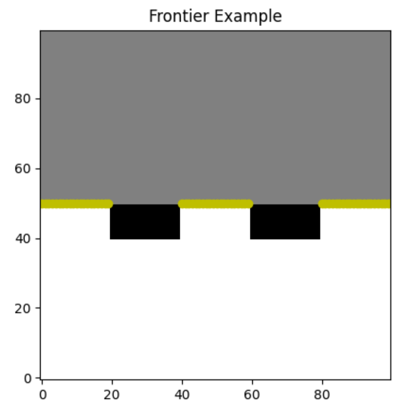
We deliberatively construct a scenario in which by one agent not exploring, the system is able to maximize information gain.

IV. Frontier Clustering

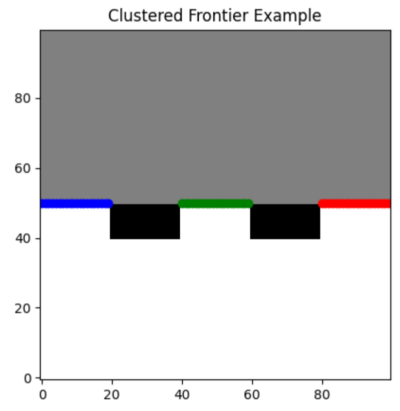
In order to determine information gain, I introduce a frontier clustering algorithm, which groups frontier cells together based on their proximity to each other. From there, the centroids of each frontier is calculated, and the expected information gain can be assigned as a function of the number of cells in the frontier.

Now, rather than assigning agents to individual cells, they can be assigned to the centroid of each frontier. In future iterations and improvements, this could be extended to reduce the complexity of the optimization problem by potentially orders of magnitude, by reducing the number of variables.

Consider the following map, with frontiers shown in yellow, unexplored areas in gray, explored areas in white, and detected obstacles in black.



Clearly, there exist two distinct frontiers. By applying DBSCAN (Density-Based Spatial Clustering of Applications with Noise) to the frontier cells, we can group them together, and identify centroids of each frontier.



Using this approach, we can reformulate the optimization problem to assign explorer agents to frontiers.

This was the problem I ended up working on in the end, as I simply did not have time in the scope of this project to complete work on everything outlined up to this point.

I instead streamlined the problem to focus on finding some kind of approximation to the optimization problem outlined above. I propose the following greedy algorithm:

IEEEabrv, main.bib

Algorithm 1 Greedy Frontier Assignment

```
1) Perform frontier clustering using DBSCAN
2) Using Djikstra's, compute the shortest path for each
   agent to each frontier centroid
3) While there are unassigned agents:
     a) Assign the agent closest to a frontier centroid to
        that frontier
     b) Check if that assignment violates connectivity con-
        straints
     c) If it does, move this agent to the back of the queue
     d) If not, remove that frontier from the list of frontiers
```