Heterogenous Autonomous Robotic Exploration (HARE)

Jackson Parker

***Abstract*—The Heterogenous Autonomous Robotic Exploration system (HARE) is meant as a proof of concept for cooperative heterogenous systems that divide tasks based on the differentiable capabilities of the robots.**

***Index Terms*—IEEE, IEEEtran, journal, LaTeX, paper, template.**

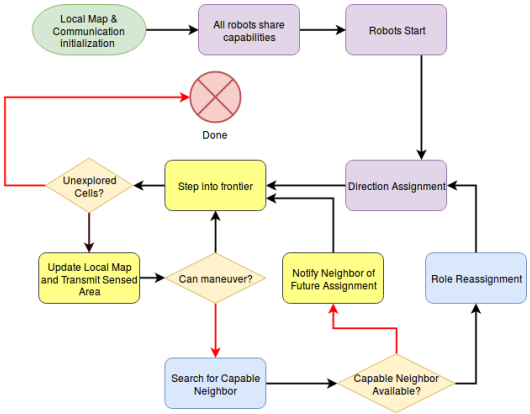


Figure 1: High level block diagram for HARE

I. Introduction

A. Overview

As an individual working in a cooperative operation, knowing the capabilities of you and your teammates is necessary to the division of tasks. This idea provides the basis for this proposed research. As homogeneous multi-robot systems are inherently limited based on the capabilities of the specific robot, it makes sense to venture down a heterogeneous avenue. A lot of research has gone into heterogeneous multi-robot systems in the past few years, but a common trend amongst most of them is that the model is built for the specific platforms that are to be used. This proposed research is meant to provide the groundwork for a heterogeneous model that allows for the insertion of a robots with different skill sets with the purpose of enhancing the overall capability of the system. This framework would allow for a variety of models to be built on top of it, with a variety of hardware. Researchers wanting to build a specific system could allocate funds and resources to build each member to accomplish a portion of the overall task, possibly allowing for more cost efficiency and remove unnecessary redundancy. The main challenge in this research will be finding the optimal way to reassign tasks based on defined capabilities of each robot in the system. This brings up a secondary, more basic, challenge of defining robot's characteristics in a simple yet comprehensive manor. All-in-all, this research acts as a proof of concept for task division based on member attributes, or individual robots capabilities, and will allow for a high degree of heterogeneous utilization and multi-robot model expansion.

B. Background

The basis of this research is in the definition of attributes which is detailed in the methodology section. This section is meant to detail previous research that will be used in proving that the fundamental idea of task division based on individual capabilities. As this research is meant to strictly provide a basis for heterogeneous task division and capability utilization, the model built on top of it could take many forms. In this case, a multi-robot behavior tree will be constructed to guide operations and define command sequences at a high level like proposed in [3] which is an expansion of research in [2]. To implement such a system the DSL buzz [6] will be used which provides a level of abstraction that allows for focus on

high level cooperation, information sharing, and simple behavioral definitions. The researchers that developed this language have created a ROS package that will be utilized, called ROSbuzz, in this proof-of-concept system. Control mechanisms and communication in multirobot systems have been a center of research for a while and updates/innovation in this corner of the field are practically constant. The Robot Operating System (ROS) [4] provides a large variety of packages that can be put together to form fully functioning and unique control mechanisms as well as providing easy to use communication packages. Exploration and path finding are widely researched and implemented using ROS packages, as components are pretty broad and can help illuminate edge cases in multi-robot systems as agents have to operate without global knowledge of their environment.

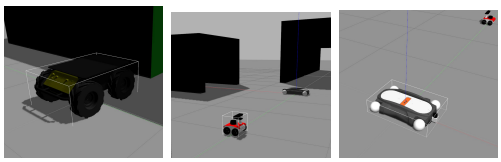
II. Methodology

A. Approach

roll assignment with algorithms

B. Role Assignment

As stated, the purpose of this research is to prove that subtask division can be accomplished based on knowledge of robot capabilities. To prove this, we are developing an exploration system in which robots with different hardware and capabilities will work together to accurately explore and map a complex region of space. Through the definition of individual attributes and the sharing of those attributes, each robot will search and map regions of space that it is capable of maneuvering. Instead of simple binary mapping of the region, 1 = free space and 0 = obstacle, each robot will mark the region that it has explored and can maneuver through this a specific identifier. This gives the region itself a bit of information as well as allows other robots to help determine if they are capable of maneuvering as well. Once a robot comes to an obstacle or region that it cannot maneuver around, it will evaluate which team members are capable of doing so and mark that space for the capable robots to explore. Due to time limitations on this project, instead of relying on complex perceptive capabilities to evaluate which robots could explore that region, the utilization of QR codes on unique obstacles will illustrate attribute sets necessary for that exploration.



(a) Husky (b) Rosbot (c) Youbot

Figure 2: Three commonly used robots with very different shapes and sizes.

As an individual working in a cooperative operation, knowing the capabilities of you and your teammates is necessary to the division of tasks. This idea provides the basis for this proposed research. As homogeneous multi-robot systems are inherently limited based on the capabilities of the specific robot, it makes sense to venture down a heterogeneous avenue. A lot of research has gone into heterogeneous multi-robot systems in the past few years, but a common trend amongst most of them is that the model is built for the specific platforms that are to be used. This proposed research is meant to provide the groundwork for a heterogeneous model that allows for the insertion of a robots with different skill sets with the purpose of enhancing the overall capability of the system. This framework would allow for a variety of models to be built on top of it, with a variety of hardware. Researchers wanting to build a specific system could allocate funds and resources to build each member to accomplish a portion of the overall task, possibly allowing for more cost efficiency and remove unnecessary redundancy. The main challenge in this research will be finding the optimal way to reassign tasks based on defined capabilities of each robot in the system. This brings up a secondary, more basic, challenge of defining robot's characteristics in a simple yet comprehensive manor. All-in-all, this research acts as a proof of concept for task division based on member attributes, or

individual robots capabilities, and will allow for a high degree of heterogeneous utilization and multi-robot model expansion.

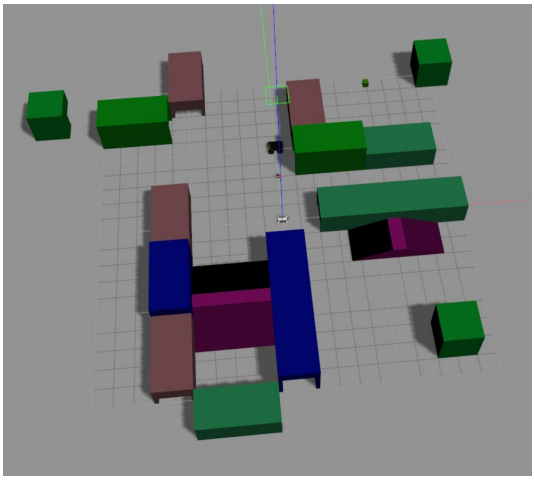


Figure 3: This is a caption of this figure.

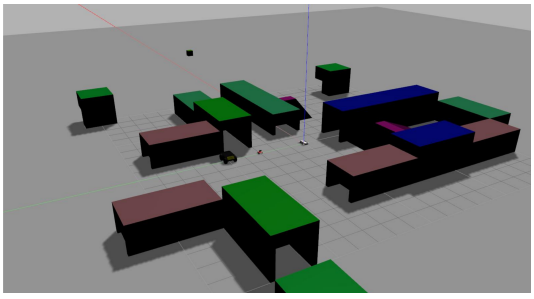


Figure 4: This is a caption of this figure.

C. Algorithms

For navigation we used two methods. The first is a Breadth First Search (BFS) which explores the cells adjacent to the robots current positions, before exploring the second level of adjacent cells. After determinining explorable cells method getPath as described in Algorithm 1 will compute the a hueristic based on the manhattan distance to determine a set of cells that make up an optimal path. For robot exploration getPath can be called with the goal being the nearest global unexplored node[**?**].

Algorithm 1 Breath First Search

```
1:  procedure GETPATH(start, goal)
2:      knownap[i][j].initializeToZero()
3:      q ← initializeQueue()
4:      q_n ← initializeQueue(init_x, init_y, 0)
5:      while q! = empty do
6:          loc = {curr.x, curr.y}
7:          if (loc = goal) then
8:              break                          ▷ reached goal
9:          h_map[curr.x, curr.y] = curr.h
        ▷ store element before popping
10:         q.pop()                            ▷ pop last element
11:         if isInBounds(loc + step) then
        ▷ check you are in map bounds
12:             test ← loc + step
13:             if isValid(test)&cellEmpty() then
        ▷ check cell unexplored and in bounds
14:                 adj ← loc + step, h + 1
        ▷ update hueristic
15:                 q.push(adj)                ▷ add to queue
16:     temp ← orderedQ(q)                     ▷ order elements based on hueristic
17:     return ← temp                          ▷ order elements based on hueristic
```

D. Multi-Agent Testing Framework

After exploring a set of options to test this idea, the best place to go was to the drawing board. Taking a step back it was decided that pushing to get straight into hardware testing was a poor decision and the underlying concept of heterogeneous task reallocation based on capabilities needed to be tested at the lowest of levels. To do this, the creation of a testing framework in roscpp was necessary. This framework allows for easy addition of robots by simply adding the details for the new robot in the xml robots launch file. To add any additional information yaml files can be used to set rosparams, just as was done here for robot capabilties. By configuring the launch file with the proper robots, one can edit the one robot launch file to have each robot launch any set of nodes that would exist in each robot's names-

pace. This allows for testing of a completely decentralized algorithm in a centralized system. The reason this was done was because of the difficulty of getting a multimaster architecture working within gazebo.

III. Results

A. Exploration

B. Role Reassignment

IV. Conclusion

A. Discussion

Even though the research.....

B. Future Work

Even though the testing framework was not the overall goal of this research, it turned out to be one of the strong suits of the project. Future work would certainly include iteration on this framework to ensure that it is as easy as possible to randomly generate maps as well as add any number of any kind of robot. The benefit of doing this on Gazebo is the ability to emulate robotic agents much closer to the real world representation than Unity, ARGoS, or any other open source platform capable of multi-robot simulation.

Another step would ofcourse be iteration of the tested cooperation model on hardware. This would require much more extraneous simulation to ensure that all edge cases are covered. To properly do this, the testing framework would have to be iterated as described above. Due to Gazebo not inherently supporting multimaster network architectures, there is the possibility that a plugin or new simulation package would need to be developed.

Appendix A
Using The Code From This Paper

The code written for this paper is available at https://github.com/uga-ssrl/hare.

Acknowledgment