Autonomous multi-agent collaborative environment exploration

Luo, Tianze

2019

Luo, T. (2019). Autonomous multi-agent collaborative environment exploration. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/136784>

<https://doi.org/10.32657/10356/136784>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 19:06:01 SGT

NANYANG TECHNOLOGICAL UNIVERSITY

Autonomous Multi-agent Collaborative Environment Exploration

A Thesis

Submitted to the School of Computer Science and Engineering
of the Nanyang Technological University

by

Luo Tianze

in partial fulfillment of the requirement for the degree of
Master of Engineering

2019

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

23 August 2019

.....
Date

Luo Tianze

.....
Luo Tianze

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

23 August 2019

.....
Date

.....
Tan Ah-Hwee

Authorship Attribution Statement

*(B) This thesis contains material from [1] paper published in the following peer-reviewed journal(s) / from papers accepted at conferences in which I am listed as an author.

Chapter 4 is published as T. Luo, B. Subagdja, D. Wang, A. Tan, "Multi-Agent Collaborative Exploration through Graph-based Deep Reinforcement Learning", 2019 IEEE International Conference on Agents (ICA), Shan Dong, China, October 18-21, 2019. (Accepted as a full paper)

The contributions of the co-authors are as follows:

- Prof Tan Ah-Hwee provided the initial project direction and edited the manuscript drafts.
- I prepared the manuscript drafts. The manuscript was revised by Dr Budhitaba Subagdja, Dr Di Wang and Prof Tan Ah-Hwee.

23 August 2019

.....
Date

Luo Tianze

.....
Luo Tianze

Abstract

Exploring an unknown environment with multiple autonomous agents is one of the fundamental research in mobility agents and is essential for numerous environment-related applications, such as autonomous cleaning, mowing, deploying, etc. The major challenge of multi-robot environment exploration is how to achieve effective collaboration so that the overall exploration strategy is efficient. One of the typical approaches is the frontier-based exploration, in which the agents move to the appropriate frontier point to explore the environment. Furthermore, exploration through partitioning maps is another efficient explore strategy.

However, although many exploration methods have been proposed, autonomous exploring an unknown environment by multiple agents still remains a difficult task. The main issue is due to the inefficient collaboration, e.g. the agents repetitively explore regions which have been explored by other agents. If many agents explore a certain area with few agents exploring other areas, the collaboration is inefficient. In contrast, allocating agents into separate areas of the map, with each agent exploring a separate area, can achieve highly efficient collaboration in the environment exploration task.

Since efficient map exploration method can highly benefit numerous related applications and there exists a large gap in multi-agent collaborative exploration, in this report, we focus on developing more efficient multi-agent collaborative environment exploration methods. We propose an efficient and robust map segmentation method, and furthermore, propose an exploration method based on the segmentation algorithm. In addition, we apply reinforcement learning on this task, and propose a novel graph-based multi-agent deep reinforcement learning method in order to derive a more efficient and scalable environment exploration strategy.

To evaluate the performance of our methods, we compare with the state-of-the-art map segmentation methods and environment exploration methods. The experimental results show that the map segmentation method can achieve more accurate map partitioning compared to the state-of-the-art segmentation methods. Based on the comprehensive understanding of the map through segmentation, our segmentation based exploration method can also achieve faster and more efficient exploration than the state-of-the-art exploration method. The graph-based multi-agent reinforcement learning method provides a special point of view on solving multi-agent environment exploration problem, and the results show that through this method, the agents are able to learn a better exploration strategy than non-learning methods.

Acknowledgements

I would like to express my sincere appreciation to my supervisor Dr. Tan Ah-Hwee. Under his kind supervision, I have learning not only substantial knowledge about multi-agent system, reinforcement learning, semantic segmentation, but also an inspiring and down-to-earth research attitude.

I would like to express my gratitude to my family for providing me with unfailing support and continuous encouragements throughout my entire life.

I would also like to acknowledge my friends and colleagues: Dr. Budhitama Subagdja, Dr. Di Wang, Dr. Jair Weigui Zhou, Mr. Pateria Shubham, Ms. Haiyan Yin, Mr. Jianda Chen, Mr. Longkai Huang, Mr. Shangyu Chen, Mr. Yu Chen, Mr. Tianbo Li, Mr. Qiu hao Zeng, Mr. Shangyu Chen, Dr. Qiang Zhou, Dr. Wenya Wang, Dr. Jiangjun Zhao, Ms. Zichen Chen and Mr. Jingzhe Zhou, for their great support and encouragement throughout my graduate study period in SCSE NTU.

Contents

Abstract	i
Acknowledgments	iii
List of Figures	vii
List of Tables	x
1 Introduction	1
1.1 Background and Motivations	1
1.2 Issues and Challenges	2
1.3 Problem Statement and Objective	3
1.4 Approach and Methodology	4
1.5 Summary of Contributions	6
1.6 Report Organization	6
2 Literature Review	7
2.1 Map Segmentation	7
2.2 Multi-agent Environment Exploration	8
2.2.1 Segmentation Based Exploration	9
2.2.2 Perception Based Exploration	9
2.2.3 Topological Based Exploration	10
2.3 Reinforcement learning	11
2.3.1 Deep Reinforcement Learning	11
2.3.2 Multi-agent Reinforcement Learning	12
2.3.3 Graph-based Reinforcement Learning	12

3	HAC-based Multi-Agent Map Exploraiton	14
3.1	Background and Motivations	14
3.2	Problem Statement	15
3.3	HAC Map Segmentation	16
3.3.1	HAC Architecture	16
3.3.2	HAC Method	20
3.3.3	Merge Fragmentary Segments	21
3.4	Map Segmentation Experiment	21
3.4.1	Dataset	22
3.4.2	Implementation	23
3.4.3	Results and Analysis	24
3.5	HAC for Multi-agent Environment Exploration	27
3.5.1	Hierarchical Adaptive Clustering Based Exploration	27
3.5.2	Real-time Clustering Based Exploration	32
3.6	Methods for Comparison	32
3.6.1	Frontier-based Multi-agent Map Exploration	33
3.6.2	Segmentation-based Multi-agent Map Exploration	35
3.7	Experimental Environment	35
3.8	Evaluation Metrics	36
3.9	Environment Exploration with Map Information	36
3.9.1	Implementation	37
3.9.2	Results and Discussion	38
3.10	Unknown Environment Exploration without Map Information	38
3.10.1	Results and Discussion	39
3.11	Exploration Under Unknown Noisy Environment	42
3.12	Conclusion	42
4	Multi-agent Graph-based Deep Reinforcement Learning for Map Exploration	46
4.1	Background and Motivations	46
4.2	Problem Statement	47

4.3	Multi-agent Centralized Learning with Decentralized Action Model . . .	48
4.3.1	Graph Construction by HAC Method	49
4.3.2	Multi-agent Graph-based DQN	50
4.4	Experiment	52
4.4.1	Environment	53
4.4.2	Implementation	53
4.4.3	Experiments on Baseline Methods	54
4.5	Results and Discussion	55
4.5.1	Performance Comparison (RQ1)	56
4.5.2	Movement Trace (RQ2)	56
4.5.3	Comparison with Decentralized Learning Model (RQ3)	57
4.5.4	Knowledge Adaptation	59
4.6	Conclusion	60
5	Conclusion & Discussion	62
5.1	Summary of The Work Done	62
5.2	Future Work	64
	References	66

List of Figures

3.1	3.1a: The ART architecture. 3.1b: The HAC architecture consisting of two modified ART neural networks: low-level and upper-level ART. Each ART consists of F_1 layer (input) and F_2 layer (category) (F_1' and F_2'). .	17
3.2	The map segmentation procedure by HAC. (a) Randomly sampled 7000 points on the empty space of the map. (b) Groups of low-level clusters wherein each group represents a segmented region. (c) The output of upper-level clustering in which each blue point represents the segment center. A graph is constructed by connecting the segment centers. (d) The final segmentation output after the fragmented segments are merged.	19
3.3	Examples of comparing hierarchical SOAS map segmentation with four state-of-the-art segmentation methods. From left to right column: (1) the ground truth; (2) Morphological segmentation; (3) Distance-based segmentation; (4) Voronoi graph-based segmentation; (5) Feature-based room segmentation; (6) HAC.	25
3.4	The model of hierarchical adaptive clustering, consisting of three layers, namely <i>input layer</i> , <i>lower layer</i> and <i>upper layer</i>	28
3.5	Figure (a) to (d) shows the procedure of hierarchical adaptive clustering: (a) sampling points. (b) <i>lower layer</i> clusters (c) <i>upper layer</i> clusters. (d) Topological information output of final clusters. Figure (e) to (h) shows the procedure of incremental segmentation during exploration: (e) sample points on the newly explored area. (f) and (g) Hierarchical clustering, which are similar to (b)(c). (h) Compute new segment centres and merge them with the segment graph.	34

3.6	Average travel distance reduction for hierarchical adaptive clustering based multi-agent map exploration under partially known map with nodes' weight and without nodes' weight, compared with the frontier based multi-agent exploration method. (b) shows the corresponding average travel distance. (The error bars and shadows indicate the standard error.)	39
3.7	Average travel distance reduction for hierarchical adaptive clustering based exploration compared with Morphological, Feature, Voronoi and Distance Transform based exploration method. (The shadows indicate the standard error.)	40
3.8	Illustration of HAC-based exploration by four agents in an unknown environment. The dark blue area indicates unexplored (unknown) area, while the gray area indicates explored area. (a), (b) and (c) represent different exploration stages, while (d) illustrates the motion directions of four agents at stage (b).	40
3.9	Average travel distance reduction for real-time clustering based exploration under the unknown map, compared with the frontier-based multi-agent exploration. (The error bars indicate the standard error.)	41
3.10	(a). Average travel distance reduction for real-time hierarchical clustering based exploration compared with frontier based exploration method on maps with different complexities. Figure (b) and (c) shows the corresponding travel distance in the simple and complex environments.	43
3.11	(a) Original map. (b) Map with 5% random noise as obstacles. (The error bars indicate the standard error.)	44
3.12	Average travel distance reduction for real-time clustering based exploration compared with frontier-based exploration method in noisy settings. (The error bars indicate the standard error.)	45
4.1	The processing pipeline of the proposed method: The system first takes an indoor situation map as input, and extracts a graph-based representations. The graph-based reinforcement learning network then takes the graph structure as the input and outputs the preferred exploration target for the agent.	49

4.2	MAG-DQN	58
4.3	Hungarian	58
4.4	Graph-based frontier	58
4.5	Motion trace of four agents during the exploration, (a) MAG-DQN, (b) Hungarian method, (c) Graph-based frontier method.	58
4.6	The performance of the centralized learning MAG-DQN compared with decentralized learning MAG-DQN with one to ten agents. The distance travelled are averaged over the last 100 episode among the 800 episodes training on the maps.	59
4.7	Apply knowledge from three agents on systems with more agents.	60
4.8	Knowledge adaptation by a large team with six to ten agents. We take the highest performance knowledge from teams with two to five agents, and apply the knowledge to larger teams with six to ten agents.	61

List of Tables

3.1	Average Precision and Recall ($\pm$ Standard Deviation) of the Segmentation Methods on 30 Maps	24
4.1	The average distance travelled by each agent during the exploration under the three maps. We compare the multi-agent Graph Q-learning with the frontier based method, the graph-based frontier method and the Hungarian method. The numbers in boldface indicate the best performance with the same number of agents.	57

Chapter 1

Introduction

1.1 Background and Motivations

Today with the advancement in the field of Artificial Intelligence (AI), intelligent robots are being used in scenarios wherein physical human intervention is deemed dangerous. For instance, in case of a forest fire, intelligent robots can be used to search for any trapped human beings and subsequently assist in their rescue by reliably communicating the victims' locations. A direct result of such a use case has been the propulsion of environment exploration problems to the forefront of mobile robotics. Environment exploration involves the deployment of either a single agent or a system of agents to traverse through an unknown environment. It is one of the essential part of robotic missions [13] such as planetary exploration [4], reconnaissance [33], rescue [48, 57], mowing [35], and cleaning [21, 36, 55].

As an essential underlying method for robotics related applications, the multi-agent environment exploration has contributed to various robotics related applications. However, though has been studied for decades, the multi-agent environment exploration, as an NP-hard problem, remains a difficult task. Comparing the single-agent map exploration, the major difficulty of multi-agent exploration is how to achieve efficient collaboration during exploration. For Instance, if too many agents crowd into the same area, while

few agent explores other area, the overall efficiency is low. In contrast, a good environment exploration strategy can separate agents into different areas in order to improve each agents exploration efficiency and reduce the interference among the members of the agent team.

Proposed by Burgard et al. [13], the frontier-based multi-agent map exploration method provides a common solution to coordinated multi-agent environment exploration tasks. In this method, each robot shares a common situation map, which includes the explored area and other agents' location. A cost-utility function is applied to navigate the agents to the most appropriate frontier points. Through this approach, during the environment exploration, each agent moves to a proper frontier point and after arrival, the agent finds the next frontier to continuously explore the new area.

A more sophisticated method, called Voronoi segmentation-based method, proposed by Wurm et al. [61], introduces the segmentation based multi-agent map exploration. The method can partition the map into multiple pieces and allocating agents to separate part of the map to perform exploration. The map partition can provide effective guidance for agent task allocation, i.e. allocating agents to different region based on the partition.

Motivated by such applications, as multi-agent environment exploration is an essential underlying method for robotics application, we focus on developing efficient and robust multi-agent environment exploration algorithms, and furthermore, benefit numerous robotics applications.

1.2 Issues and Challenges

A core challenge in multi-agent environment exploration, different from the single agent exploration, is how to achieve effective and efficient multi-agent collaboration. The collaboration is mainly a task allocation mechanism, which allocates agents with individual exploration tasks, such as regions to explore.

Without an efficient collaborative mechanism, the overall exploration can be extremely inefficient. As for multi-agent system during the exploration, if too many agents explore the same area, with few agents exploring other areas, the overall efficiency is quite low. Interference among agents should also be taken into consideration, as when too many agents crowd into an area, the interference among sensors and traffic jam also cause low efficiency.

In contrast, strategically allocating agents into separate regions is a good approach to tackle the multi-agent environment exploration problem. The agents, therefore, explore separate areas individually without interfering with other agents or repeatedly exploring the area that is explored by other agents.

1.3 Problem Statement and Objective

The objective of this report is to explore a novel, efficient, accurate and robust environment exploration method for multiple agents autonomously exploring an unknown indoor environment.

We consider the map as the grid map, where each cell of the grid represent each of the following values:

- Obstacle: indicate an obstacle presents in the cell.
- Empty space: indicate the cell is empty.
- Agent: indicate an agent takes up the cell.

The indoor map normally contains multiple rooms and corridors. The specific configurations of the map will be shown in Chapter 3.

To compare the performance of our methods with the existing multi-agent environment exploration methods, we constructed a simulator using Python as our simulation

environment. We use the average travel distance among the agent team as the performance metric. The shorter the average travel distance, the better the method performs. The measure of exploration time spent is another commonly used indicator, which commonly applied to real robots' performance. As the simulation time highly depends on the computer, programming language and implementation, the results may not fully reflect the performance of the method itself. Thus, in this report, we mainly apply the average travel distance as the performance indicator for every multi-agent environment exploration method.

1.4 Approach and Methodology

Based on the idea that allocates agents to separate regions, firstly, we design a novel and accurate map segmentation method, called the Hierarchical Adaptive Clustering (HAC) method. The HAC method is built on hierarchical Adaptive Resonance Theory (ART) architecture [28], and provides a hierarchical clustering method to group near-by regions into a common segment. The HAC method performs map segmentation in a bottom-up manner through the hierarchically clustering approach.

Furthermore, based on the HAC method for map segmentation, we propose a new real-time incremental and iterative task allocation method for multi-agent exploration in an indoor environment using HAC map segmentation and cost-utility measures to allocate the exploration tasks for each agent. The hierarchical segmentation method is based on the HAC method, where randomly sampled points from the map allowing it to conduct incrementally and iteratively in real-time to come up with a set of segments. Based on the segmented map, an exploration task allocation method is applied to assign every agent to a segment to visit based on a cost-utility measure. The cost measures the distance from the selected agent to a target location, and the utility measures the potential of a target location to be explored by any agent. By applying the cost and

utility function, we not only consider the cost of visiting the targeted location by an agent, but also consider the potential discovery of new areas during the exploration.

The HAC-based map exploration method serves under two different situations: exploration based on known environments (map of the environment is available at the beginning) and exploration in unknown environments (the map is constructed as a part of the exploration process). The major difference of the strategies under both conditions is the segmentation method. Under the known environment, we segment the whole map once at the beginning of the exploration; while under the unknown environment, we perform real-time incremental segmentation.

In addition, we apply reinforcement learning techniques on this task to achieve a more efficient environment exploration strategy through an autonomous learning process, instead of hand-crafted policies. As the HAC method provides a sophisticated segmented map representation of the environment, we extract a graph representation of the segments from the segmented map, where each vertex of the graph represents a segment, and each edge represents the distance between two corresponding segments. We design a graph-based multi-agent reinforcement learning method for the graph of the map, such that the agents are to learn their exploration strategy upon the graph instead of the raw map.

We compare the proposed methods with other state-of-the-arts segmentation-based exploration methods of task allocation for indoor floor plan Voronoi segmentation, distance-transform segmentation, feature-based segmentation, and morphological segmentation for task allocation in known environments. We also compare it with Frontier-based exploration method as the baseline to deal with unknown environments.

We quantitatively and qualitatively analyze the experiment results for our proposed method and state-of-the-art methods. The results show that our proposed method can achieve high efficiency and robustness in performing multi-agent environment exploration.

1.5 Summary of Contributions

The main contribution of this report are listed as follow:

- Based on the Adaptive Resonance Theory (ART) model, we propose a hierarchical adaptive clustering (HAC) method to semantically segment a map, and furthermore, we design the HAC-based multi-agent map exploration method allowing a team of agents efficiently exploring an unknown environment.
- We propose a multi-agent graph-based reinforcement learning method, such that through the learning process, the learning model can construct high efficient exploration strategies for multi-agent environment exploration.

1.6 Report Organization

The reminders of the report are written as follow. Chapter 2 presents the literature review of map segmentation, multi-agent environment exploration and deep reinforcement learning. Chapter 3 presents the HAC method for map segmentation and the HAC-based multi-agent map exploration method. Chapter 4 presents the multi-agent graph-based reinforcement learning method for map exploration. The last chapter concludes the report and discusses the contribution presented in this report.

Chapter 2

Literature Review

2.1 Map Segmentation

Semantically segmenting indoor floor plans or maps into meaningful regions is an important task for many robotic related applications[9]. An indoor environment such as the floor plan of a building can be segmented into meaningful regions for different purposes. For example, environment or map segmentation can be applied on mobile robots as support for exploration task in a search and rescue mission [52], or to plan efficient floor cleaning strategies for cleaning robots [8] or to assist a mobile robot teleoperator [25].

Meyer and Beucher introduces the Morphological Segmentation method [44], which performs on a grid-based map in which cells of the grid map are initially labeled as accessible or not accessible by agents. With the morphological erosion operator, the walls of the map iteratively grow by one pixel in each step. The method uses connected component analysis to verify whether connected accessible areas have become separated until all accessible cells have been removed.

Distance Transform-Based Segmentation is another commonly used algorithm. The transform is the distance of an accessible pixel to the closest border pixel. The local maximum always lies in the centre of a room, and it is smaller than those inside a large room with corridors or doors. Segmentation completes until the room found centre is labeled uniquely and extended into remaining unlabeled space via wavefront propagation.

Feature-based Segmentation is described in the work by Óscar Martínez Mozos et al. [50]. The data for feature computation is collected by simulated rays. From the scan, a set of simple geometric features has been computed. The feature vector is classified by an AdaBoost classifier [24] into room labels. The AdaBoost classifier can further segment new environments.

The Voronoi graph-based segmentation is the best performing method among the state-of-the-art methods [9]. It firstly computes a voronoi graph on a grid map, and gets the critical points which have exactly two closest obstacle cells, from the graph. The critical lines, which are the lines connecting the critical points and the closest obstacles, are drawn to separate the map into voronoi cells. Finally, the voronoi cells are merged to form a segmented map according to a set of pre-defined rules.

The Voronoi graph-based segmentation method, though it achieves the best performance among these methods [9], it requires a set of manually defined rules to guarantee the performance. The morphological segmentation and distance transform-based segmentation also require manually set parameters to secure the performance. In this paper, we compare our proposed segmentation method with these 4 state-of-the-art methods.

Another popular method in recent year is Computer Vision Based Segmentation. Dodge et al. [20] use fully convolutional networks to train the floor plan dataset to detect the wall border and exploit Faster R-CNN to detect room labels.

2.2 Multi-agent Environment Exploration

Studies on multi-agent exploration have emerged in recent decades. Different techniques and approaches have been used like the frontier-based method wherein each agent makes its own decision to select a target to visit though they share a global world representation [62]. Bautin et al. improved the frontier-based exploration by ranking the agents to allocate to a particular frontier location based on their travel distances to the frontier [6].

Other recent variants of frontier-based exploration on multi-agent exploration include a distributed multi-robot model [41] and some others involving communication constraints and collaboration among the agents [2, 3, 16].

2.2.1 Segmentation Based Exploration

The segmentation based approach reduces the spatial complexity of the environment, departing the environment to different individual areas. A typical method is called a Voronoi diagram, which specifies a set of obstacles points and segments the areas with equidistant from closest obstacles. Wurm et al. [61] presents a strategy to solve the coordinated multi-agent exploration problems, which is based on Voronoi segmentation to assign the agents tasks [61]. Voronoi diagram is used to produce a set of one-dimensional connecting nodes, representing different tasks for agents. The nodes and edges of the Voronoi diagram have a maximal distance of at least two points of the obstacle points in the map. It collapses leave edges into the node points to prune to the major skeleton and connect the critical points. Because of Voronoi diagram mechanisms, the potential issue is it produce trivial segmentation easily. Exploration with segmentation may not lead the shortest path but tends to perform maximal clearance of the unexplored areas. Other works based on the map segmentation include Morphological operators, distance transform-based method and feature-based method [9]. Those works are mainly for specific targets, and their performances are with high variance, which reduces the exploration efficiency.

2.2.2 Perception Based Exploration

Perception mapping is a qualitative method to address this problem, with a semantic description of exploration surroundings. Beetz et al. describes the task with perception method is using natural language, which is a human-like socialized function [7]. The

perception way is similar to human-being to relate the signs and things with their meaning. Thus, perception method is aiming to identify the meanings of exploration areas, performing humanoid decision behaviours. During the environment exploration, the perception information can be hierarchically stored with the relationship of the entities. Classification of rooms, corridors, and doorways need to be organized to exploration graph, which is a way to represent the spatial information. Zender et al. designed a method to construct conceptual spatial representations for indoor mobile robots to boost the indoor environment exploration. The method creates spatial and functional properties of typical indoor environments as the concepts to help the robots recognize and understand the environment. The perception of the semantic mapping method comprises high-level features, modelling the human esthesia competency and trying to relate the connections of observation [43].

2.2.3 Topological Based Exploration

Multi-agent exploration problem can be considered as an instance of the optimization problem, with the trade-off of exploration cost and area utility [13], such that the agents are to achieve accurate environment exploration with minimal time spent. It is an NP-hard problem [26]. The topological method directly creates a topological map with map cells that represent spatial area information. Topological based approach regularly can be used with a hierarchical exploration construction. The approach described in [67] employs graph partitioning to achieve a hierarchical path planning method. More precisely, the hierarchical exploration architecture can be considered as basic navigation graph and simplifying this graph with graph cut criteria. Similar work for exploration with the incremental map is presented in [11], in which the authors propose a method based on the online generation of topological maps and compute the graph partitioning with spectral clustering.

Additionally, Govindarajan et al. utilize the topological based method to solve the human-agent collaborative exploration problem, which is more challenging than a pure multi-agent environment [27]. The exploration problems are likely to have noise and errors because of the dynamical process. Thus, the approach based on the topological is a proper way to make it robust to environment noises and measurement errors.

2.3 Reinforcement learning

2.3.1 Deep Reinforcement Learning

Deep reinforcement learning has been applied to solve challenging problems in recent years, such as robotics control [29, 66], game playing [45, 46] and financial data analysis [19]. As for multi-agent map exploration, though people have developed many solutions to solve this problem as illustrated in Section 2, no polynomial time solutions have been found to get the optimal solution to this problem. Therefore, through deep reinforcement learning approach, it is possible that an agent team may learn a well-performing model based on interaction with the environment.

The General Reinforcement Learning Architecture (Gorila) [49], which performs asynchronous training of reinforcement learning agents in a distributed setting. In Gorila, each process contains an actor that acts in its own copy of the environment, a separate replay memory, and a learner that samples data from the replay memory and computes gradients of the DQN loss [46] with respect to the policy parameters.

Following the success of DQN [22, 46], several well performing models have been well formulated, such as Actor-critic [5], Asynchronous Actor-critic [42, 47], Deep Deterministic Policy Gradient (DDPG) [40], Proximal Policy Optimization (PPO) [53] for single agent as well as multi-agents reinforcement learning.

2.3.2 Multi-agent Reinforcement Learning

One intuitive approach to train a multi-agent system is to apply independent learning agents, such that each agent learns the knowledge by itself without sharing with other agents [42]. The main difference between the multi-agent system and the single agent system is that in the multi-agent system, the agents usually need to cooperate or compete with each other in order to achieve good learning outcomes. Proposed by Tampuu et al. [56], the DQN is used to solve multi-agent learning in cooperative and competitive environments. In this setting, though agents use separate neural networks for individual and decentralized learning, a well-structured reward function is designed to lead the agent learning cooperative manners.

A centralized learning with decentralized action model for multi-agent reinforcement learning is proposed by [42], achieves better performance in cooperation and competition environment. In this method, the actor-critic model is used, where for each agent, the action selected by the actor is decentralized, while, the policy update through the critic not only considers the environment, but also takes into account other agents actions. Through this approach, the agents are able to learn a cooperative strategy in order to achieve more rewards as a team.

2.3.3 Graph-based Reinforcement Learning

Following the success of Convolutional Neural Networks (CNN) in image processing, recent research on graph pattern recognition aims to formulate analogies of CNN on graphs for applications such as pattern recognition, object detection, and social network analysis, etc. [18, 32, 37, 39]. The first trial of formulating the graph convolution procedure was accomplished by Bruna et al. [10], in which the authors propose both spatial and spectral convolution methods on graphs. The spatial convolution method performs multi-scale

clustering of the graph, while the kernel for convolution is over-localized and hard to be adapted to match varying local neighbourhoods.

The spectral graph convolution was constructed based on the spectral graph theory [15]. The method defines the smoothness function as the basis for computing the eigenvectors and the eigenvalues of graph Laplacian L , and identify the spectral filters attached to eigenvalues. The idea of the spectral graph convolutional networks was extended by Defferrard et al. by proposing the fast localized and polynomial parameterized filters for performing graph convolutional operations [18]. This method extends the radius of the localized filters, i.e., from 1 hop to K hops. In this report, we adopt the graph convolutional neural network approach [18] and extend it to address graph-based multi-agent map exploration problem.

Though few works have been done on graph-based reinforcement learning, Dai et al. proposed "S2V-DQN", which is a graph-based reinforcement learning method integrating "structure2vec" and DQN [17]. The "S2V-DQN" can achieve relatively good results on some complicated graph problem such as single agent Euclidean TSP problem. However, the multi-agent map exploration problem can be similar to multi-agent non-Euclidean TSP problem, which has not been solved by the current graph based reinforcement learning methods.

Chapter 3

HAC-based Multi-Agent Map Exploraiton

In this chapter, we present the architecture and algorithm of HAC, followed by the HAC-based multi-agent map exploration method. We also provide corresponding experiments to evaluate the performance of the HAC method and the HAC-based multi-agent environment exploration method for both map segmentation and environment exploration.

3.1 Background and Motivations

Exploring an enclosed environment by a team of autonomous agents has remained a challenging problem. In various domains like search and rescue [63], cleaning robots [60], reconnaissance [63], or sensors deployment system [59], initial information or prior knowledge relevant to the task or the model of the environment can be partially or totally unavailable. The agents must incrementally construct a model or a map representation of the environment in order to plan and assign the exploration tasks to the right agents. What the agent should do or where it should go to visit next can only be determined after the map is constructed at least partially.

The major issue of the multi-agent environment exploration is how to achieve effective coordination among the team of agents. In order to achieve efficient and fast environment

exploration, the agents need coordination in order to avoid repetitively exploring the regions that explored by other agents, and encourage each agent to explore separate regions of the map.

Although existing methods can achieve a certain level of coordination, there still exists a large gap in achieving effective coordination and efficient exploration. Therefore, we propose the HAC-based multi-agent exploration method, which can allocate exploration tasks for each agent based on semantically partitioning the environment, to provide more efficient coordination and exploration strategy for the multi-agent system in performing environment exploration.

3.2 Problem Statement

Suppose we have a grid-based map, where each cell of the map represents each of the following values:

- Obstacle: indicate an obstacle presents in the cell.
- Empty space: indicate the cell is empty.
- Agent: indicate an agent takes up the cell.

We have N number of autonomous agents, which can move freely on the empty space of the map. Each agent equips a 360° laser sensor with a detection radius r cells, such that the agent can detect its surrounding within r cells during exploration. Each agent can move one cell forward in one step and keeps moving during the exploration.

As long as there exists any unexplored area, all the agents continuously explore the environment until the whole environment is explored. The target is to explore the entire environment with minimum average traveling distance of the agent team.

3.3 HAC Map Segmentation

The HAC method provides a novel map segmentation algorithm as well as architecture. The key idea of HAC is that close areas are clustered together into common regions in a bottom-up manner.

3.3.1 HAC Architecture

The HAC architecture is built of two different modified Adaptive Resonance Theory (ART) [14] neural networks, as shown in Figure 3.1b. The ART is a self-organizing neural network model, which can cluster the input samples into different categories [14]. An example of the ART architecture is shown in Figure 3.1a.

The HAC architecture utilizes two different ART model, denoted as the low-level ART, which is placed at the bottom, and the upper-level ART, which is placed at the top. In the HAC algorithm, we firstly take many sample points from the map, where for each sample, we can know the position and the property of that sample, where the property denotes whether the sample belongs to obstacles or empty spaces. The sample points are the inputs to the HAC model, and the low-level ART clusters the sampled points into low-level segments, where the F_2 layer outputs the cluster centre of the low-level segments. The upper-level layer takes the F_2 layer as the input and groups the low-level clusters into larger segments. The ART model we use in this architecture is the ART-2A-ES, which is a new variant of the network for handling partitioned points and regions in Euclidean space.

3.3.1.1 Low-level ART Network

The low-level ART neural network is shown in the bottom-half part of the Figure 3.1b. The sampled point's position is denoted as $p = (x, y)$ in an $M \times N$ two-dimensional map. The normalized point position $\mathbf{p} = (p_1, p_2)$ is used as an input to the ART network by

(a) An Example of ART Architecture

(b) The architecture of proposed HAC

Figure 3.1: 3.1a: The ART architecture. 3.1b: The HAC architecture consisting of two modified ART neural networks: low-level and upper-level ART. Each ART consists of F_1 layer (input) and F_2 layer (category) (F_1' and F_2').

normalizing p is used as the input vector to the low-level ART model, where $p_1 = \frac{x}{M}$ and $p_2 = \frac{y}{N}$ so that $p_1, p_2 \in [0, 1]$.

We denote $\mathbf{P}$ as the set of all normalized points on the map, where $p \in P$. In ART 2A-ES, a cluster node j in F_2 layer is activated based on a bottom-up euclidean choice function.

$$T_j = 1 - \|\mathbf{p} - \mathbf{w}_j\|_2 \quad (\text{Eq. 3.1})$$

where $\mathbf{p}$ is the input vector in F_1 (normalized node), and $\mathbf{w}_j$ is a weight vector associating the input F_1 with the node j in F_2 . $\|\mathbf{p} - \mathbf{w}_j\|_2$ denotes the euclidean distance between $\mathbf{p}$ and $\mathbf{w}_j$. For every input sample p and cluster node j , a choice value T_j is assigned according to the complement of euclidean distance between $\mathbf{p}$ and $\mathbf{w}_j$, which is j 's central point.

The proper node J is selected to categorize $\mathbf{p}$ in F_2 if the *resonance* condition is satisfied, such that

$$T_J = \max\{T_j \mid \text{for all } j \text{ in } F_2, T_j \geq \rho\} \quad (\text{Eq. 3.2})$$

where $\rho \in [0, 1]$ is the *vigilance* parameter. If no F_2 node satisfies the *resonance* (for every j , $T_j < \rho$ always holds), a new *uncommitted* node is recruited in F_2 to represent the new input. Thus, the clusters are growing with novel input comes.

In ART-2A-ES, a particular weight update operation is applied whenever a node J is selected so that $\mathbf{w}_J$ is updated as follows,

$$\mathbf{w}_J^{(\text{new})} \leftarrow \frac{m^J \cdot \mathbf{w}_J + \mathbf{p}}{m^J + 1}, \quad (\text{Eq. 3.3})$$

where m^J is the number of input points categorized as J so far. This contrasts with the conventional update function used in other ART networks [14, 23]. Here, the weight $\mathbf{w}_J$ represents the center (centroid) of the cluster rather than the node's matching template.

In this approach, a point p is considered to be in a category j if its normalized position ($\mathbf{p}$) has the closest Euclidean distance towards j (located at $\mathbf{w}_j$) but cannot be greater than ρ as the *resonance* criteria. From empty areas on the map, n points are randomly sampled to make the set of sampled points $P = \{p_i\}_{i=1}^n$. Each $p_i \in P$ is presented consecutively as an input to be clustered or grouped as a node J . Thus, each node j in F_2 represents a segment in the map with a center point $\hat{p}^j = (\hat{x}^j, \hat{y}^j)$ as the inverse normalization of its weight vector $\mathbf{w}_j$.

Figure 3.2: The map segmentation procedure by HAC. (a) Randomly sampled 7000 points on the empty space of the map. (b) Groups of low-level clusters wherein each group represents a segmented region. (c) The output of upper-level clustering in which each blue point represents the segment center. A graph is constructed by connecting the segment centers. (d) The final segmentation output after the fragmented segments are merged.

3.3.1.2 Upper-level ART network

The center point $\hat{p}^j$ of each low-level segment node j in F_2 is then presented one at a time as the input to the F_1' layer of the upper-level ART to generate the upper-level segments or clusters in F_2' (Figure 3.1). This upper-level ART works similarly as the low-level one. However, a modified *choice function* is applied to every low-level cluster's central point to generate the upper-level clusters so that any pair of the upper-level cluster and low-level input cluster with a non-empty point (e.g door, wall, partition) in between will be excluded in the *resonance* search process. The upper-level choice function is as follows,

$$T_j^{(\text{Upper})} = \begin{cases} 0 & \text{if there exists } \mathbf{p}' \in \mathcal{P}^c \text{ so that} \\ & \|\mathbf{p}' - \mathbf{w}_j\|_2 + \|\mathbf{p} - \mathbf{p}'\|_2 = \\ & \|\mathbf{p} - \mathbf{w}_j\|_2 \\ 1 - \|\mathbf{p} - \mathbf{w}_j\|_2 & \text{otherwise.} \end{cases} \quad (\text{Eq. 3.4})$$

In this case, $T_j = 0$ if there is a point $\mathbf{p}'$ in the set of occupied points $\mathcal{P}^c$ (the complement of empty areas $\mathcal{P}$) located along the straight line between $\mathbf{p}$ and $\mathbf{w}_j$. Based on the selected cluster and the input, its central point can be obtained by applying the centroid-based weight update function.

3.3.2 HAC Method

Specifically, the entire hierarchical segmentation procedure is summarized as follows.

Step 1. Detect doors as well as small connections between segments based on the Shi-Tomasi corner and door features detector [54, 58]. This detector identifies small opening areas and narrow corners to be excluded from empty areas ($\mathcal{P}$) or to be considered as occupied regions or obstructions.

Step 2. Randomly sample n points from all empty areas on the map as set P .

Step 3. Generate lower-level segments $\mathcal{S}_{\mathcal{L}}$ by presenting every point p_i in P as the input to the low-level ART and cluster them into category nodes in F_2 layer based on low-level *vigilance* ρ as the closeness criteria.

Step 4. For every segment j in $\mathcal{S}_{\mathcal{L}}$, present its center point $\hat{p}^j$ as the input to upper-level ART in which the same process of *resonance* search is applied to form larger segments based on the upper-level *vigilance* ρ^u as the closeness criteria. By presenting $\hat{p}^j$ of every segment j in $\mathcal{S}_{\mathcal{L}}$, the set of upper-level segments $\mathcal{S}_{\mathcal{U}}$ can be generated.

Step 5. For every segment k in $\mathcal{S}_{\mathcal{U}}$, all the points $P_{\mathcal{U}}^k$ of the segment k include all points in low-level segments that belong to segment k so that $P_{\mathcal{U}}^k = \bigcup_{j=1}^r \hat{P}^j$ in which segments 1 to r in low-level ART are included in k . The center points of low-level segments and all the points for each upper-level segment are illustrated in Figure 3.2 (b) and (c) respectively.

3.3.3 Merge Fragmentary Segments

Some segmented regions in a particular area such as a large room or corridor in the map can still be merged together to form a larger meaningful chunk of the segment after the clustering process by the upper-level ART network. In this matter, let G be a graph representation wherein each node (vertex) corresponds to a segment's centre with edges connecting to every other node of the other segments. Each edge in G is associated with a value indicating the Euclidean distance between the connected nodes.

Through the *edge-cutting* process, the graph G can be trimmed such that any edge with a larger value (longer) than threshold δ or if there is an obstruction in between can be removed. Thus, the trimmed graph can then be decomposed into several sub-graphs. All connected segments in a sub-graph are merged to form a new larger segment with a new segment point based on its geometric center (centroid). The result of this graph-based segments merging and partition process is shown in Figure 3.2 (d).

The overall algorithm of HAC is illustrated in **Algorithm 1**.

3.4 Map Segmentation Experiment

In this section, we present a series of experiments of indoor map segmentation among HAC and different state-of-the-art methods. As elaborated at the beginning of this chapter, the state-of-the-art methods for comparison are the morphological method, feature-based method, distance transform-based method and voronoi segmentation method.

Algorithm 1: HAC Algorithm

Input : area $\mathcal{A}$ in the map**Output:** The list of segments with the corresponding centroids

- 1 Pre-process $\mathcal{A}$ for door and door features with Shi-Tomasi approach
 - 2 Sample n points $P = \{(x_i, y_i)\}_{i=1}^n$ on $\mathcal{A}$
 - 3 **for** each p_i in P **do**
 - 4 Normalize p_i into input vector $\mathbf{p}$ to present to lower level network in F1
 - 5 Perform clustering with lower-level network to generate the set of cluster center points $\mathcal{S}_{\mathcal{L}}$ in F2
 - 6 **end**
 - 7 **for** each $\hat{p}^j$ in $\mathcal{S}_{\mathcal{L}}$ **do**
 - 8 Put (normalized) $\hat{p}^j$ as the input to upper level network in F1'
 - 9 Perform clustering with upper-level network (excluding points with obstruction in between the cluster center) to generate the set of cluster center points $\mathcal{S}_{\mathcal{U}}$ in F2'
 - 10 **end**
 - 11 Construct graph G based on $\mathcal{S}_{\mathcal{U}}$
 - 12 Perform graph cut on G to identify corridors and big rooms
-

3.4.1 Dataset

We conduct the map segmentation experiments based on 30 floor plans extracted from the publicly available datasets. Among these samples, 20 maps are taken from the IPA rooms segmentation dataset, which is indoor map dataset containing 20 office floor plans collected by Bormann et al. [9]. The original sources of the IPA room segmentation dataset are the Radish Repository [34] and the Oscar Martinez Mozos dataset. The other 10 maps we use in these experiments are taken from the R-FP dataset [20], which is a Japanese indoor floor plan dataset.

The maps are converted into grid-based maps, where each cell of the grid map represents either the obstacle or empty space. In addition, we take the human label as the ground truth. Since the IPA room segmentation dataset provides human labeled ground truth, we directly utilize those labels as our ground truth for map segmentation. Meanwhile, we manually label the R-FP dataset, since the data doesn't associate with the segmentation labels in the dataset.

3.4.2 Implementation

We implement the HAC method with the following configurations:

- Low-level vigilance ρ : 0.05.
- Upper-level vigilance ρ^u : 0.2.
- Number of sampled points n : 7000.

As the Morphological, Voronoi, distance-based and feature-based methods are implemented by Bormann et al. [9] and are publicly available, we apply these implementations on our dataset. We apply the default configuration and parameter settings of these methods.

To evaluate the performance of the HAC method, we apply the implementation of the state-of-the-art map segmentation methods for comparison, including the Morphological segmentation method, distance-based segmentation method, Feature-based segmentation method and the Voronoi segmentation method.

Voronoi Segmentation: Voronoi based method provides another appropriate approach to solve the multi-agent exploration problem [61]. It distributes the agents to structure regions which are partitioned into segments. The generation of Voronoi segments can be described that the nodes and edges of Voronoi graphs have a maximal distance of at least two points of the obstacle points in the map, and the Voronoi graph collapses leave edges into the node points to prune to the major skeleton and connect the critical points. The advantage is it applied Voronoi diagram function to segment unexplored environment to some meaningful regions.

Morphological Segmentation: Meyer and Beucher introduce the Morphological Segmentation method, which works on a grid map in which pixels are initially labeled accessible only at mapped accessible areas [44]. With the morphological erosion operator,

the walls of the map are grown iteratively by one pixel. Using connected component analysis to verify whether connected accessible areas have become separated until all accessible cells have been removed.

Distance Transform Segmentation: The transform is the distance of an accessible pixel to the closest border pixel. The local maximum always lies in the centre of a room, and it is smaller than those inside a large room with corridors or doors. Segmentation completes until the room found centre is labeled uniquely and extended into remaining unlabeled space via wavefront propagation.

Feature-based Segmentation: Óscar Martínez Mozos et al. mention that the data for feature computation is collected by simulated rays [50]. From the scan, a set of simple geometric features has been computed. The feature vector is classified by an AdaBoost classifier [24] to predict room label.

3.4.3 Results and Analysis

The results of map segmentation by the related methods are shown in Figure 3.3. We evaluate the performance of each method by comparing with the human-labels, which are shown in the leftmost column of Figure 3.3. From column two to column five shows the Morphological segmentation method, Distance-based method, Voronoi method and the Feature-based method. The rightmost column shows the results generated by the HAC method.

Table 3.1: Average Precision and Recall ($\pm$ Standard Deviation) of the Segmentation Methods on 30 Maps

	Morphological	Distance Transform	Voronoi	Feature-based	HAC
Precision	0.96 $\pm$ 0.04	0.96 $\pm$ 0.04	0.94 $\pm$ 0.04	0.91 $\pm$ 0.10	0.97$\pm$0.03
Recall	0.84 $\pm$ 0.13	0.89 $\pm$ 0.10	0.94 $\pm$ 0.07	0.81 $\pm$ 0.21	0.95$\pm$0.04
F-measure	0.90	0.93	0.94	0.85	0.96

We quantitatively analysis the segmentation results among different method using a standard metric of segmentation, which includes *Precision*, *Recall* and *F-measures* [9].

Figure 3.3: Examples of comparing hierarchical SOAS map segmentation with four state-of-the-art segmentation methods. From left to right column: (1) the ground truth; (2) Morphological segmentation; (3) Distance-based segmentation; (4) Voronoi graph-based segmentation; (5) Feature-based room segmentation; (6) HAC.

$Precision = \frac{tp}{tp+fn}$, where tp denotes the true positive, which is the area that is correctly segmented by the segmentation method coincident to the ground truth. fn denotes the false negative, which is the incorrect segmented area of the segmented map comparing to the ground truth. The *precision* represents containment of the generated segments in all areas of the ground truth. $Recall = \frac{tp}{tp+fn}$ denotes the containment of the ground truth in all areas of the generated segments.

Precision is high if the segmented rooms are fully contained in the ground truth rooms, while recall is high if the ground truth segments are fully contained in the segmented rooms. Explicitly, high precision with low recall indicates the segmentation method tends

to over-segment the map, while high recall and low precision indicates the segmentation method tends to under-segment the map. In addition, $F - measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$ is a combination of precision and recall measures. Therefore, using F-measure can comprehensively evaluate the performance of segmentation methods.

From the results, we can interpret that HAC method achieves best segmentation results among the 30 real-world challenging map. Morphological method, distance transform method and the feature based method achieves a high score in precision but a low score in recall, indicating that these methods tend to over-segment the map. In another saying, these methods tend to segment the map into too many small pieces, but fail to extract the complete segment from the raw maps.

As mentioned before, autonomously exploring an environment through agents is one of the fundamental research in the autonomous agents field. Meanwhile, numerous related applications are built upon the research, such as mowing robots, office cleaning robots [60], reconnaissance [63] and autonomous deployment [59].

Although many autonomous environment exploration methods have been developed, the map exploration by autonomous agents remains a difficult task. As for a single agent, the task is to explore the environment fast and robust. However, when it comes to the multi-agent system, the task becomes complicated, as the coordination among the team becomes more and more important as the number of agents grows.

In the next chapter, we discuss the state-of-the-art multi-agent map exploration methods, including the frontier-based methods, and the segmentation based methods. Based on the idea of map segmentation and the segmentation research presented in Section 3.3, we develop the HAC-based multi-agent map exploration method. The HAC-based method is built upon the methodology of HAC map segmentation, in which, through partitioning the map into multiple separate regions, the agents are assigned to the proper region to perform map exploration task.

Through this approach, we develop a map exploration method under two different scenarios. The first scenario is when the floor plan of the map is given but the environment remains unknown so that the agents need to explore the unknown environment based on the given floor plan. The second scenario is that the floor plan of a map is not given so that the map needs to be explored from scratch by multiple autonomous agents.

Furthermore, we compare the HAC-based multi-agent map exploration method with state-of-the-art exploration methods, including the frontier-based method and segmentation-based methods. The results show that the HSAOS method can explore the map with minimum time comparing to other state-of-the-art methods.

3.5 HAC for Multi-agent Environment Exploration

In this section, we present the HAC-based multi-agent map exploration method, including its architecture, procedures and the detailed algorithms. Our multi-agent environment exploration method can work in two different scenarios: the partially known environment and unknown environment. Under partially known environment, the layout of the map is given as the prior knowledge for environment exploration, and the agents may explore the environment according to the layout. While, in the unknown environment, no layout information is given to the agents before exploration.

Similar to the Hierarchical self-organizing adaptive segmentation method, the HAC architecture of the HAC-based multi-agent exploration method consists of two modified Adaptive Resonance Theory (ART) model. The architecture is shown in Figure 3.4.

3.5.1 Hierarchical Adaptive Clustering Based Exploration

The multi-agent system is a hierarchical centralized system, comprising a commander and a group of individual agents. The commander acts as the brain of the system and has the global information to decide on the exploration strategies for the agent team, while

Figure 3.4: The model of hierarchical adaptive clustering, consisting of three layers, namely *input layer*, *lower layer* and *upper layer*.

each agent has local area exploration capabilities. Typically, the commander allocates agents to different regions, and agents explore maps within its regions. The overall goal of the system is to complete exploration of the map with minimum travel distance. We define the cost and utility function for agent allocation that each agent selects the optimal region with the trade-off of utility and cost during the exploration.

Cost: The cost of each agent is a proper way to measure the distance between the current agent to possible target regions. When exploring the environment, each agent has many potential targets, as the interesting regions. If the commander directly sends the agent to such unexplored targets, this will result in some repeated exploration, decreasing the performance quality. Additionally, there are many aspects involved to address multi-agent exploration problem with interaction during the exploration. Thus, decision strategies with global situation considerations are particularly relevant to achieve successful multi-agent exploration.

Exploring the environment with multiple agents raises the problem of how to assign the target regions to individual agents. Given the fact that a multi-agent exploration task is a complex system, it is necessary to trade off all-round aspects so that we can

avoid several agents moving to the same regions. To figure out the issues mentioned before, we consider cost (distance) firstly as one of the strategy influence factors. To determine the cost of the current region cluster, we compute the optimal path from the agents' position to central points of clusters. Based on the A* search algorithm [30], we can efficiently find the optimal path and calculate path distance on a 2D grid map. Based on path distance $d_{A^*_{r_k}, t_i}$, which defines the path distance from agent r_k to target point t_i , we define the cost of a target point as Equation Eq. 4.1.

$$C_{t_i, r_k} = \left\{ d_{A^*_{r_k}, t_i} \mid i \in \{1, \dots, n\} \right\}. \quad (\text{Eq. 3.5})$$

where C_{t_i, r_k} is cost of agent r_k to target point t_i , and since in this method, agents are assigned to different clustering regions of the map, t_i represent the center of the target cluster.

Utility: In the multi-agent exploration problem, it is difficult to predict the individual decision of each agent at every step and the decisions are easily affected by the surrounding environment[61]. Thus, the other significant consideration of multi-agent exploration is how to send agents to different regions, reducing the possibility of the same targets.

Given the situation that one agent has moved to a particular region, the other agents should not explore this region if there are still unexplored regions. We consider utility as a measurement to predict the agents' decision. The utility of a cluster U_i indicates the value for assigning the current agent to cluster S_i . Intuitively, if a cluster is near to an agent, the utility for other agents to explore this cluster should be lower, since the agent can easily complete exploring this region. Furthermore, when an agent moves to a cluster, the adjacent utilities should decrease to prevent other agents to come to this area. We define the initial utility for all clusters to be 1, and during exploration, when an

agent reaches a new cluster, we adjust the surrounding segments' utilities as the following equation:

$$U_i = \begin{cases} \max \left(U_{\min}, U_i - \left(1 - \frac{d_{A^*r_k, S_i}}{\beta} \right) \right), & d_{A^*r_k, S_i} < \beta \\ U_i, & \text{otherwise} \end{cases}$$

where U_i is the utility of cluster S_i , $d_{A^*r_k, S_i}$ is the path distance from agent r_k to cluster S_i , and β is an adjustable parameter to define the range of affected clusters.

Weight: During the multi-agent map exploration, intuitively, the border region of a map should be explored preemptively comparing to a centre region if they have same cost and utility since a central region is more likely to be explored by other agents. We propose the weight parameter for each cluster node to obtain the additional information of the possible targets, which is represented the global exploration structure for the agents. The calculation of weights are based on Google power iteration: $x_{k+1} = (1 - \mu)Ax_k + \mu e$, $k \in \mathbb{N}_0$ [12]. Through the power iteration, the sequence of eigenvectors $\{x_k\}_{k \in \mathbb{N}_0}$ converges to the importance score $(e_1, \dots, e_n)$ of each node. Furthermore, we define the weight of each node as the following equation:

$$w_i = 1 - \frac{e_i}{\max_i(e_i)}$$

where $(w_1, \dots, w_n)$ represents the weight of the nodes in the graph.

3.5.1.1 Target Selection in the Partially Known Environment

To select the optimal target for individual agents, we need to consider the trade-off among the cost of moving to a target, the utility of that target and the weight of the target correlation to the global environment. Our algorithm encourage agents to explore regions with high utility, low cost and high weight. High utility score of a target indicates that there's few nearby agents to explore this target; low cost indicates the target is near

Algorithm 2: Target cluster allocation for agent r_k

1 Find cluster

$$S_{i,r_k} = \max_i (U_i - \gamma \cdot C_{i,r_k} + \sigma \cdot W_i \mid i \in \{1, \dots, m\})$$

2 Assign agent r_k to cluster S_i

3 Update the utility for the nearby clusters $\{S_j\}_{j=1}^m$,

$$U_i = \begin{cases} \max \left(U_{\min}, U_i - \left(1 - \frac{d_{A^*r_k, S_i}}{\beta} \right) \right), & d_{A^*r_k, S_i} < \beta \\ U_i, & \text{otherwise} \end{cases}$$

to the agent; and high weight indicates the target is at the border of the map, as border areas are less likely to be explored by the agents. The algorithm is summarized in **Algorithm 2**.

3.5.1.2 Target Selection in the Unknown Environment

To choose the optimal targets for individual agents, we need to balance the trade-off of different aspects. The cost to the possible targets, the utility and the weights of those targets jointly influence the decision of the commander. Additionally, multi-agent exploration in the unknown environment can be considered in two situations: $n_s \geq n_a$ and $n_s < n_r$, where n_r represents the number of agents and n_s represents the number of segments. Since the strategies for the different environment should be different to achieve the ideal performance.

In particular, when the number of unexplored clusters is greater than agents, we consider more about how to avoid individual agents exploring the same regions. To determine the appropriate target for individual agents, we compute the **Algorithm 2** to find the optimal task allocation strategy at each step. After the agent reaches the target, it will perform frontier-based exploration within its region. The agent will be assigned to the next cluster when it completes its task within the current region.

When the number of unexplored clusters is less than the number of agents, the more important we need to consider is energy-saving, which requires the individual agents making decisions based on others' actions. We first allocate the agents to occupy all the unexplored segments, while other agents will perform frontier-based exploration. Whenever new clusters are spotted, we assign unallocated agents to these clusters. Once the unexplored clusters are greater than the number of the agents, the exploration team will perform the **Algorithm 2**, in this way, the dynamic strategies lead the exploration becoming more robust and suitable for a lot of environment difference.

3.5.2 Real-time Clustering Based Exploration

In practice, properly clustering the unknown environment is crucial to complete the exploration task efficiently. For example, the clustering of the unexplored environment can be a target signal for individual agents. If there are only frontier points for agents without any additional global information, it is hard for them to take the right actions. Agents may easily explore very close frontier points, which will cause unreasonable targets allocation.

To provide a better exploration, we integrate the target clustering selection method, the hierarchical clustering method, and the frontier-based local exploration method, into a complete multi-agent exploration method under partially known map or unknown map, which is the real-time clustering based exploration. The multi-agent clustering-based exploration method is summarized in **Algorithm 3**. A demonstration of the real-time clustering during the exploration is shown in Figure 3.5.

3.6 Methods for Comparison

We compare the performance of the HAC-based multi-agent map exploration method with the state-of-the-art exploration methods, including the frontier based method, and

Algorithm 3: Exploration Task Allocation based on Cost-Utility Measure

```

1 perform segmentation with Algorithm 1 if known area  $\mathcal{A}$  is initially available
2 while exploration is unfinished do
3   for each agent  $k$  do
4     if new area  $\mathcal{A}$  is uncovered by agent  $k$  then
5       | perform segmentation in  $\mathcal{A}$  with Algorithm 1
6     end
7     for each segment  $i$  do
8       | update costs  $C_{I,K}$  and utility  $U_i$ 
9     end
10    allocate agent  $k$  to segment  $s_k$ , where  $s_k \leftarrow \arg \max_i (U_i - \gamma \cdot c_{i,k} + \sigma \cdot w_i)$ 
11    if no unallocated  $s_k$  is available (all have been allocated to other agents) then
12      | find the set of frontier points  $V$  within the segment where agent  $k$  is
13      |   currently in
14      |   allocate a frontier  $v \in V$  to agent  $k$  to visit
15    end
16 end

```

the segmentation-based method, such as the Voronoi-based segmentation, Morphological-based segmentation, feature-based segmentation and the distance transform based segmentation.

3.6.1 Frontier-based Multi-agent Map Exploration

As a traditional successful well-known method for autonomous environment exploration, the frontier-based exploration methods have been widely used in multi-agent map exploration tasks [13]. The advantage of the frontier-based method is obvious, the stability and robustness of the method can support a large number of agents under various maps with relatively stable performance. Therefore, the frontier-based exploration method is a good baseline method for comparing multi-agent map exploration methods.

According to the method developed by Burgard et al. [13], during the exploration, the agents share a common knowledge of the environment. A centralized algorithm is applied to allocate each agent to appropriate frontier point. The task allocation for each

Figure 3.5: Figure (a) to (d) shows the procedure of hierarchical adaptive clustering: (a) sampling points. (b) *lower layer* clusters (c) *upper layer* clusters. (d) Topological information output of final clusters. Figure (e) to (h) shows the procedure of incremental segmentation during exploration: (e) sample points on the newly explored area. (f) and (g) Hierarchical clustering, which are similar to (b)(c). (h) Compute new segment centres and merge them with the segment graph.

agent is based on a cost and utility function, where the utility denotes the advantages of selecting a frontier point as a target, and the cost is represented by the travel distance from an agent to the corresponding target.

The target selection for agent i based on the cost and utility function is described as follows:

$$(i, t) = \arg \max_{(i', t')} (U_{t'} - \beta \cdot V_{t'}^{i'}), \quad (\text{Eq. 3.6})$$

where $U_{t'}$ is the computed utility of target t regarding to the agent i , and $V_{t'}^{i'}$ is the travel distance to target t from agent i . β is the weight parameter for the cost. The best target for each agent is determined by maximizing the cost and utility equation in every decision cycle.

3.6.2 Segmentation-based Multi-agent Map Exploration

As illustrated in Chapter 2, the idea of allocating agents based on a partition of the map can potentially improve the exploration performance. To comprehensively evaluate the performance of the HAC based method, inspired by the voronoi segmentation based exploration [61], we develop an exploration method of integrating Hungarian task allocation method [38] and the various map segmentation method. Through this approach, we implement the Voronoi segmentation based exploration, morphological segmentation based exploration, feature-based segmentation based exploration, and distance-transform segmentation exploration method.

3.7 Experimental Environment

To evaluate the HAC-based multi-agent exploration performance, we conduct a series of environment exploration experiments on 10 real-world challenging maps, compared with the state-of-the-art methods which are mentioned in Section 4.2. The 10 maps are taken from the ROS indoor map collection [9], which are widely used in related research. Each of the 10 maps contains more than 20 rooms, and as the maps are from the real world and not easy for exploration, they are quite suitable to test the performance of multi-agent map exploration algorithms.

The map are pre-processed to grid-based map, where each cell of the map consists one of the following status:

- **Explored:** This cell is explored by at least one agent.
- **Unexplored:** This cell haven't been explored by any agents.
- **Obstacle:** This cell is occupied by obstacles that agents cannot pass.
- **Agent:** This cell is occupied by an agent.

In order to make the results comparable to the following related research among different maps, the maps are defined to maintain a width of 200 cells, while the height are adjusted according to the original aspect ratio.

3.8 Evaluation Metrics

To evaluate the performance of different exploration methods, we use the average travel distance across each agent within a single exploration task. Specifically, the less travel distance by each agent during the environment exploration, the better the method is. Furthermore, we assume each agent equips a 360° Lidar with a radius of $r = 15$ cells, such that in every step, the agent is able to detect the surrounding 15 cells area.

Those evaluations are used to avoid software processing speed influence. We define the travel distance reduction as

$$Reduction = (d_b - d_p) / d_b \times 100\%$$

, where d_b represent the travel distance of the baseline method, and d_p represent the travel distance by our proposed method.

3.9 Environment Exploration with Map Information

Nowadays, as the indoor map technology becomes more and more general, in many cases, the agents are able to explore the environment based on a pre-known map. For example, in the fire disaster in a shopping mall, the agents are able to perform search and rescue task according to the indoor map of the mall. Under this scenario, the task of the agents is to explore the environment with minimum time according to based on the map information.

Under this scenario in this experiment, we evaluate the HAC-based multi-agent map exploration method by comparing with the frontier-based exploration method and

the segmentation based exploration methods, including Voronoi-based segmentation, Morphological-based segmentation, feature-based segmentation and the distance transform based segmentation. Since the map is given as prior information, for the segmentation based exploration, we firstly apply map segmentation to partition the map into multiple meaningful partitions. Then, we use the Hungarian method, which is an efficient and well-known task allocation method, to allocate the robots to different regions of the map to perform exploration.

3.9.1 Implementation

We implement the HAC-based multi-agent exploration method according to the following settings:

Hierarchical clustering model

- vigilance ρ_l in lower layer: 0.05
- vigilance ρ_u in upper layer: 0.2

Multi-agent map exploration

- Cost parameter γ : 0.01
- weight parameter σ : 0.35
- Utility update parameter β : 0.6

The vigilance ρ in upper layer, which controls the size of the segment, is corresponding to the sensor range of the agents, such that when an agent is at the centre of a segment, it is able to explore the entire segment. The vigilance ρ_l controls the size of the lower layer clusters, and it should be suitable with size of the upper layer clusters. Other parameters are selected based on the experiments.

We apply the implementation [9] of the Voronoi segmentation, morphological segmentation, distance-transform segmentation and feature-based segmentation to our experimental environment.

3.9.2 Results and Discussion

The results of map exploration by HAC-based method and frontier-based method are shown in Figure 3.6.

Figure 3.7 shows the average travel distance during exploration by HAC-based method, Morphological method, feature-based, voronoi and the distance transform based method.

The quantitative results indicate that using HAC-based multi-agent map exploration method, the system can achieve the best performance compared to the state-of-the-art methods under this scenario. Due to the efficient segmentation and finding the centre of each segment, the agents are efficiently assigned to the appropriate regions. Through this approach, the overall exploration efficiency is increased.

3.10 Unknown Environment Exploration without Map Information

Another common scenario for map exploration is to explore an environment which is totally unknown, i.e. no prior knowledge of the map information is given. Therefore, the robots are to explore the environment while constructing the map. Under this scenario, we construct the real-time segmentation based map exploration method. The key idea of this method is to conduct the map segmentation while performing map exploration. The segmentation is designed to be incremental segmentation, in which only the newly explored area is to be segmented, while the area that is already segmented will remain the same. The algorithm can decide whether the newly explored area belongs to a new region or should be categorized into some existing segments.

(a)

(b)

Figure 3.6: Average travel distance reduction for hierarchical adaptive clustering based multi-agent map exploration under partially known map with nodes' weight and without nodes' weight, compared with the frontier based multi-agent exploration method. (b) shows the corresponding average travel distance. (The error bars and shadows indicate the standard error.)

3.10.1 Results and Discussion

Figure 3.8 shows the procedures of four agents exploring an unknown environment using the HAC-based method. As shown in Figure 3.8 (d), during the exploration, the agents are able to separate to explore regions as the HAC method provides an incremental segmentation on the current explored area.

The experiment results of average travel distance gain for HAC-based method under unknown environments comparing to the frontier-based method are shown in Figure 3.9.

Figure 3.7: Average travel distance reduction for hierarchical adaptive clustering based exploration compared with Morphological, Feature, Voronoi and Distance Transform based exploration method. (The shadows indicate the standard error.)

Figure 3.8: Illustration of HAC-based exploration by four agents in an unknown environment. The dark blue area indicates unexplored (unknown) area, while the gray area indicates explored area. (a), (b) and (c) represent different exploration stages, while (d) illustrates the motion directions of four agents at stage (b).

(a)

(b)

Figure 3.9: Average travel distance reduction for real-time clustering based exploration under the unknown map, compared with the frontier-based multi-agent exploration. (The error bars indicate the standard error.)

From the results, we can interpret that through an incremental segmentation of the map, the HAC method can achieve superior performance than the frontier-based multi-agent map exploration method. Though based on limited map information, the incremental segmentation can still maintain a superior understanding of the map features than purely cost-utility function by the frontier-based method.

We furthermore test the exploration results under the environments with different complexity. The sample maps are partitioned into two types according to the number of rooms contained by the map:

- maps with the simple environment,
- maps with the complex environment.

As the number of rooms grows, the map becomes more and more complicated for autonomous agents' exploration. Therefore, testing the exploration strategies under different complexity environment is a good way to comprehensively evaluate the performance of that strategy.

The results of map exploration under the two types of maps are shown in Figure 3.10. No matter in simple environments or complicated environments, the HAC-based exploration method maintains a superior performance than the frontier-based exploration method. Meanwhile, under complicated environments, the coordination strategy of our proposed method becomes more and more effective in achieving efficient exploration than in simple environments.

3.11 Exploration Under Unknown Noisy Environment

In order to test the robustness of the approach in more realistic settings, we introduce 5% random noise to the environment as false detection of empty space, such that during the exploration, the sensors get imperfect detection of the environment due to the falsely-detected obstacles. Sample maps with the original and noisy settings are shown in Figure 3.11. Figure 3.12 shows the results under unknown environments. The experiment with the noisy settings shows high robustness of the proposed exploration method. Despite the errors to some extent in identifying empty areas, the overall task of exploration can still be conducted more efficiently than the baseline frontier-based method. The performance is also scalable in terms of the number of agents.

3.12 Conclusion

In this chapter, we discuss the map segmentation problem and multi-agent map exploration problem, and we propose the HAC method as well as HAC-based multi-agent map

(a)

(b)

(c)

Figure 3.10: (a). Average travel distance reduction for real-time hierarchical clustering based exploration compared with frontier based exploration method on maps with different complexities. Figure (b) and (c) shows the corresponding travel distance in the simple and complex environments.

Figure 3.11: (a) Original map. (b) Map with 5% random noise as obstacles. (The error bars indicate the standard error.)

exploration method to tackle the problems.

The HAC method can provide a fast, accurate and robust map segmentation in a bottom-up manner with the hierarchical ART model. Based on the segmented map by HAC, the HAC-based multi-agent map exploration method can efficiently allocating exploration tasks to agents to achieve efficient map exploration.

The HAC method can also provide incremental segmentation, which only segments the exploration area. This property highly supports the multi-agent exploration in an unknown environment, such that the HAC can provide a sophisticated representation of the explored area, and we can, therefore, allocate agents to explore different regions based on the incremental segmentation.

(a)

(b)

Figure 3.12: Average travel distance reduction for real-time clustering based exploration compared with frontier-based exploration method in noisy settings. (The error bars indicate the standard error.)

Chapter 4

Multi-agent Graph-based Deep Reinforcement Learning for Map Exploration

4.1 Background and Motivations

Convolutional Neural Network (CNN) provides a good solution for highly structured data-driven pattern recognition, such as image classification, speech recognition, deep reinforcement learning to play Atari games [1, 31, 46]. Notably, CNN is characterized by its strong ability of hierarchically extracting important features from a given input. For example, when using deep reinforcement learning for playing the Atari games, CNN takes in a continuous sequence of screenshots as the state input and produces the extracted features of the current state. The extracted features are then fed into a series of fully connected layers, which outputs the action policy to control the game. Since CNN is quite powerful and has significant generalization ability, the CNN based deep reinforcement learning can achieve and even exceed human level players in playing Atari games and controlling robots, etc. Proposed by Zhu et al., the CNN based reinforcement learning for map exploration has been used for the single agent map exploration prob-

¹This chapter is published as **T. Luo**, B. Subagdja, D. Wang, A. Tan, "Multi-Agent Collaborative Exploration through Graph-based Deep Reinforcement Learning", 2019 IEEE International Conference on Agents (ICA), Shan Dong, China, October 18-21, 2019. (Accepted as a full paper)

lem in relatively simple environment [65]. However, for multi-agent map exploration in a complex environment, the state space for the agents may be very large. As such, learning may become quite inefficient and time-consuming. In extreme cases, the agents are even unable to thoroughly explore the environment, due to the huge exploration space.

In this chapter, we present a graph-based highly coordinated multi-agent reinforcement learning method to address the multi-agent map exploration problem. Based on the situation map, we construct a graph representation of the environment, wherein each node of the graph represents a region on a map, a room, or a part of a corridor etc, and each edge in the graph represents the path and the distance between two regions. Furthermore, we adapt the Deep Q Network for learning with the graph-based representation and construct a graph-based multi-agent centralized learning model named MAG-DQN, for the multi-agent map exploration problem.

We test our method by comparing with the grid frontier based coordinated multi-agent map exploration algorithm [13], as well as the graph-based methods, including the graph-based frontier and Hungarian method. The experimental results show that our proposed method can provide a feasible solution to the multi-agent map exploration problem. Furthermore, our proposed model exhibits good generalization ability across agent teams of different sizes.

4.2 Problem Statement

Suppose we have a grid-based map, where each cell of the map represents each of the following values:

- Obstacle: indicate an obstacle presents in the cell.
- Empty space: indicate the cell is empty.
- Agent: indicate an agent takes up the cell.

The HAC method, as introduces in Chapter 3, can generate a graph representation G of the map through map segmentation. In the graph G , each vertex (node) represents a segment centre while each edge between two vertexes (nodes) represent the distance between two segment centres in the map.

Suppose we have N number of autonomous agents, which can freely travel along the edges of the graph G from one node to another. The segment area is pre-defined by the HAC method such that when an agent moves to a segment centre, it is able to explore the entire segment.

As long as there exists any unexplored area, all the agents continuously explore the environment until the whole environment is explored. The target is to explore the entire environment with minimum average traveling distance of the agent team.

4.3 Multi-agent Centralized Learning with Decentralized Action Model

In order to construct a learning model for multi-agent map exploration problem, we propose the Multi-agent graph-based Deep Q learning (MAGDQN) to achieve better multi-agent map exploration performance.

The proposed method contains two part. The first part is constructed by hierarchical segmentation, which is similar to the HAC method presented in Chapter 3. The second part is the Multi-agent graph-based Deep Q learning (MAGDQN), which is a graph-based centralized Q-learning model for multi-agent reinforcement learning.

The flow of the method is shown in Figure 4.1. The process takes the original grid map as the input and performs HAC segmentation process to obtain the segment centres. The following step is to construct a graph based on the segment centres, where the node of the graph represents the segment centre and the edge represents the path distance between two corresponding segment centres.

Figure 4.1: The processing pipeline of the proposed method: The system first takes an indoor situation map as input, and extracts a graph-based representations. The graph-based reinforcement learning network then takes the graph structure as the input and outputs the preferred exploration target for the agent.

After the graph is constructed, the graph is to go through the MAGDQN learning process, and the final output will be the Q-values for the appropriate moving target for the corresponding agent.

4.3.1 Graph Construction by HAC Method

To provide a good graph representation of the map, we utilize the HAC method to perform segmentation on the map to get the segment centres, and thereafter, build the graph representation based on the segment centres. Thus, in the graph, the nodes represent the segment centre and the edges represent the path distance from the corresponding two segment centres.

As mentioned in Chapter 3, the HAC method performs the hierarchical segmentation on a map. After two layers of hierarchically clustering, the segment centres are generated as the final output in this procedure. The edge cut merges segment process is omitted in this algorithm, since for the exploration, only small segments are needed such that the agent can explore the whole segment when the agent is at the segment centre.

Therefore, through this process, the graph can maintain a good topological represent the map. The environment exploration task on the grid map can be, therefore, trans-

formed into a graph exploration task. In the graph, each node of the graph is assigned to one of the following value:

- Unexplored: the node is not explored by any agents.
- Explored: the node has been explored by an agent.
- Current agent: the node contains my current agent.
- Other agent: the node contains the other agent/agents.

Sparsity. The graph is designed to be sparse, such that the edge of the graph is omitted is the normalized edge value (path distance) is beyond a certain value. The process follows the following equation.

$$W_{i,j} = \begin{cases} \frac{d_{i,j}-d_{min}}{d_{max}-d_{min}}, & d_{i,j} \leq \lambda, \\ 0, & d_{i,j} > \lambda. \end{cases}$$

where the $W_{i,j}$ is the weight of the edge $E_{i,j}$ after the process. If the original value of the edge is greater than or equal to λ , the weight of that edge is set to zero.

4.3.2 Multi-agent Graph-based DQN

After the graph generation process, we can perform the graph reinforcement learning process to train the agent team for environment exploration with the following processes.

Graph Laplacian. The normalized graph Laplacian matrix is calculated as follow:

$$\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}$$

Where $\mathbf{I}$ is the $n \times n$ identity matrix and $\mathbf{D}$ is an $n \times n$ diagonal matrix where $\mathbf{D}_{i,i} = \sum_j \mathbf{W}_{i,j}$.

Spectral graph CNN Since the graph Laplacian L is a real positive semi-definite matrix, we can compute the set of orthonormal eigenvectors $\{u_i | i \in [1, n]\} \in \mathbb{R}^n$ through

eigen-decomposition $L = U\Lambda U^T$, where $U = [u_0, \dots, u_{n-1}] \in \mathbb{R}^{n \times n}$ is the set of eigenvectors, also known as the Fourier basis for the graph Laplacian. $\Lambda = \text{diag}([\lambda_0, \dots, \lambda_{n-1}]) \in \mathbb{R}^{n \times n}$ is the diagonalized real nonnegative eigenvalues, corresponding to eigenvectors.

We follow the spectral graph convolution method [18], where the localized convolutional filters are defined in Equation Eq. 4.1:

$$g_\theta(\Lambda) = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\Lambda}) \quad (\text{Eq. 4.1})$$

$T_k(x), k = [0, \dots, K-1]$ is the k th order Chebyshev polynomial, which defines the k -hop convolutional filters. θ_k are the learnable parameters with the k th order Chebyshev polynomial. $\tilde{\Lambda} = 2\Lambda/\lambda_{\max} - I_n$ is the normalized diagonal matrix of scaled eigenvalues that lie in $[-1, 1]$. The convolutional operation can be defined in Equation Eq. 4.2:

$$y = g_\theta(L)x = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\Lambda})x \quad (\text{Eq. 4.2})$$

Multi-agent deep Q-learning The deep Q-learning, also known as DQN is a famous deep reinforcement learning method, and has been applied in multiple agents environments [22, 51]. DQN usually makes use of the CNN to predict the action-value given a state input and based on the Bellman equation, the Q function can be written recursively as $Q^\pi(s, a) = \mathbb{E}_s[r(s, a) + \gamma \mathbb{E}_{a' \sim \pi}[Q^\pi(s', a')]]$. The loss function is therefore calculated by the equation:

$$\mathcal{L}(\theta) = \sum_{i=1}^n E_{(s_i, a_i, r_i, s'_i)} [r_i + \gamma \max_{a'_i} Q_i(s'_i, a'_i, \tilde{\theta}_i) - Q_i(s_i, a_i, \theta_i)]$$

.

where θ_i are the parameters in the Q network, and $\tilde{\theta}_i$ are the parameters in the target network. In this paper, we present the multi-agent graph based deep Q learning, where we use a graph as input to the networks, and we develop centralized learning and decentralized action for this model. Using centralized learning, the agents share and

contribute to the same Q network, where each agent uses the same state space. The model is designed as follows.

We consider that the system involves n agents and each agent i can obtain its state $s_{i,t}$ at time t . At each given state s for agent i , we compute the Q value by considering other agents actions $A = [a_0, \dots, a_{i-1}, a_{i+1}, \dots, a_{n-1}]$. Here, we integrate A into state s and compute $Q_i^\pi(s, a)$ for the agent i . Each agent in the team uses the same Q network, which is updated in a centralized basis by considering all the agents' performance:

$$\mathcal{L}(\theta) = E_{(s_i, a_i, r_i, s'_i, i \sim n)}[r + \gamma \max_{a'} Q_i(s'_i, a'_i, \tilde{\theta} - Q_i(s_i, a_i, \theta))] \quad (\text{Eq. 4.3})$$

The detailed multi-agent Q-learning algorithm is explained in the Algorithm 4.

Algorithm 4: MAG-DQN

```

1 Initialize experience replay memory  $\mathcal{M}$  to capacity  $\mathcal{N}$ 
2 Initialize Graph  $G$ 
3 Randomly drop  $n$  dots in the map for episode  $e = 1$  to  $M$  do
4   for step  $t = 1$  to  $T$  do
5     for agent  $i = 1$  to  $N$  do
6       Construct state  $s$ 
7        $a_{i,s} = \begin{cases} \text{random node } v \in G, & \text{w.r.t. } \epsilon \\ \text{argmax} Q(s_t, a_t) \end{cases}$ 
8       Execute action  $a$  and update state  $s_{t+1}$ 
9       Store transition  $(s_t, a_{i,t}, r_t, s_{t+1})$  to memory
10      Sample random minibatch  $B$  of transition  $(s_j, a_{i,j}, r_j, s_{j+1})$  from  $\mathcal{M}$ 
11      Update  $\theta$  by SGD for  $B$ 
12    end
13  end
14 end
15 Return  $\theta$ 

```

4.4 Experiment

In this section, we describe our experiments on multi-agent map exploration using MAG-DQN, compared to frontier-based method, graph-based greedy method and graph-based

Hungarian method.

We aim to answer the following research questions:

- **RQ1:** How does the MAG-DQN perform compared to the commonly used methods?
- **RQ2:** How do the agents achieve coordination with the MAG-DQN?
- **RQ3:** Can the centralized learning model perform better than the decentralized learning model? What is the difference of performance between the two models?

4.4.1 Environment

To evaluate the MAGDQN for environment exploration, we conduct the experiments on 3 real-world challenging maps from [9], compared with the state-of-the-art methods which are mentioned in the Section 4.2.

4.4.2 Implementation

The experiments are conducted under the following parameter settings:

Hierarchical clustering and graph building

- vigilance ρ_l in lower layer: 0.05
- vigilance ρ_u in upper layer: 0.15

ρ_u is the main parameter to adjust the size of the segments. Here we take $\rho_u = 0.15$, so that under the pre-defined sensing range, when a robot moves to a segment center, it is able to explore the whole segment.

MAG-DQN settings

- learning rate: 0.005

- Gamma: 0.7
- epsilon greedy: linear decay from 0.95 to 0.05 in 3000 episodes.
- Action: the agents are only allowed to move to its adjacent nodes in each step.
- Episode: Each episode contains at most 100 steps. If within 100 step, the agent team has not completely explored the whole graph, the episode is forced to stop.

During the exploration, each agent is only able to choose its adjacent nodes in the graph as the next movement target. If two nodes are not connected, an agent cannot directly move from one node to the other.

The evaluation metric is the same as the metric in Section 4.

4.4.3 Experiments on Baseline Methods

To evaluate the performance of MAG-DQN method, we conduct experiments with the same settings for the baseline methods: frontier-based coordinated multi-agent map exploration method [13], graph-based frontier method and graph-based Hungarian method. The frontier-based multi-agent exploration method is grid-based method and the other two are graph-based method, which use the same graph generated by our proposed HAC method.

Frontier based method.[13] The agent team shares the global information among each agent, and each agent selects its movement target based on a generally used utility function [13]:

$$\underset{(t_1, \dots, t_n)}{\operatorname{argmax}} \sum_{i=1}^n [U(t_i | t_1, \dots, t_{i-1}, t_{i+1}, \dots, t_n) - \beta(V_{t_i}^i)^2]$$

, where $U_{t_i}^i$ represents the utility of a frontier point t_i for robot i , and $V_{t_i}^i$ represents the cost of robot i travel to the frontier point t_i .

During the exploration, the agents move at a constant speed, and in each decision making cycle, an agent chooses its most favorable frontier point on the map as the target

and moves to that target. The simulation stops when the agents thoroughly explore the entire map, and the average traveling distance made by all the agents are recorded as the performance indicator: shorter traveling distance indicates better performance.

Frontier-based graph method. Since there is no well defined frontier-based exploration method for multi-agent exploration on graphs, we define the cost function for agents selection frontier nodes.

$$C(i, v_i) = \begin{cases} d(i, v_i) & \text{unexplored} \\ \infty & \text{explored} \end{cases}$$

During exploration, each agent selects the frontier nodes with least cost, i.e., the nearest unexplored node. After an agent explores a frontier node, other agents will not list this node as a target, but can still pass through the node. When the map is fully explored, we take the average travel distance for each agent as the performance measure for this method.

Hungarian method. The Hungarian method [38] is a well-known method to solve multi-agent task allocation problem, and it is often used in multi-agent map exploration problem. In this experiment, we adopt the Hungarian method to the graph-based multi-agent map exploration problem, and the method is elaborated as follows.

In each step, we use the Hungarian method to allocate agents to a unexplored node. The allocation minimizes the total travel distance for the agent team in this movement, and the objective function for each allocation is according to the optimization function:

$\min_{(t_1, \dots, t_n)} \sum_{i=1}^n [D(i, t_i)]$, where t_i is the unexplored node assigned to agent i . Through this iteration, in each step, the agent team can greedily select the current best action.

4.5 Results and Discussion

In this section, we discuss the performance of MAG-DQN, compared with the frontier-based method, the graph frontier method and the Hungarian method.

4.5.1 Performance Comparison (RQ1)

For each map with each number of agents, we conduct 20 trials, where in each trial we randomly select a starting point for the agent team. We take the traveling distance per agent as the result of the trial and average the 20 trials' results as the final results for the agent team, which are shown in Table 4.1. The experiment results show a general improvements of the exploration efficiency of our proposed MAG-DQN method comparing to the baseline methods. Although, with small agent team, the MAG-DQN method obtains longer travel distance than the baseline methods, the MAG-DQN can learn effective collaborations and outperforms baseline methods when the number of agents in a team is greater than three.

Another interesting finding is that all the graph-based methods outperform the frontier-based method. This is mainly because the graph generated by the hierarchical clustering method obtains the essential spatial information of the map, wherein the graph, each node represents a individual region and each edge represents the connection between the two regions. Therefore, with a comprehensive graph topological representation, the collaborative exploration methods can achieve better performance than the exploration on the grid map.

4.5.2 Movement Trace (RQ2)

We also analyze the coordination of the agents based on their movement trace during the exploration. Figure 4.5 shows the agents' movement traces with the MAG-DQN method, graph-based Hungarian method and graph-based frontier method. We use the movement trace of four agents in an exploration experiment to illustrate the difference in the exploration strategies of each method. In this experiment, each robot starts from the same starting point at the beginning, and only when the map is fully explored, the exploration process will stop.

Table 4.1: The average distance travelled by each agent during the exploration under the three maps. We compare the multi-agent Graph Q-learning with the frontier based method, the graph-based frontier method and the Hungarian method. The numbers in boldface indicate the best performance with the same number of agents.

Number of agents	1	2	3	4	5
Frontier	1121.26	654.15	494.42	412.03	362.95
Graph-based Frontier	846.73	454.8	370.37	308.62	276.67
Hungarian	846.73	431.36	339.38	293.52	270.26
MAG-DQN	951.23	517.23	372.83	287.83	266.91
Number of agents	6	7	8	9	10
Frontier	333.72	302.24	282.54	266.25	263.59
Graph-based Frontier	263.59	245.07	227.23	223.29	205.99
Hungarian	247.21	226.66	222.49	214.07	201.88
MAG-DQN	243.39	218.54	210.31	197.34	188.89

It can be observed that with the MAG-DQN method, the agents are allocated to explore separate regions of the environment. With MAG-DQN method, the interference among agents, i.e. the overlap of the trace, is less than the other two methods. In Figure (b) and (c), the movement trace is relatively messy and sometimes the agents need to travel a long distance to move to the desired region. Therefore, the task allocation to the agents by MAG-DQN can be observed more efficient compared to the Graph-based Frontier method and the Hungarian method. Based on the efficient task allocation, the MAG-DQN can therefore achieve better performance in terms of travel distance comparing to the baseline methods.

4.5.3 Comparison with Decentralized Learning Model (RQ3)

We also compare our centralized learning model with a decentralized learning model, wherein each robot maintains its own deep Q network and does not share the knowledge learned with the others. In this experiment, the agents start to explore the environment at a pre-defined entrance, stop when the entire map is explored, and maintain the configurations as the centralized model.

Figure 4.2: MAG-DQN

Figure 4.3: Hungarian

Figure 4.4: Graph-based frontier

Figure 4.5: Motion trace of four agents during the exploration, (a) MAG-DQN, (b) Hungarian method, (c) Graph-based frontier method.

Figure 4.6 shows the performance comparison between the centralized learning MAG-DQN with the decentralized learning MAG-DQN. The results show the average performance of last 100 episodes, where the training contains 800 training episodes. The results clearly indicate that, within a certain training episode, the performance of the centralized learning model is better than the decentralized one.

As discussed in Section (II).C, in the decentralized model every agent learn different Q values separately and differently, while in centralized learning model, all the agents learns to optimize the overall performance of the agent team. Though the decentralized model may finally converge to a similar performance as the centralized model with a large amount of training samples, in general, the performance of decentralized learning is inferior than the centralized learning model.

Figure 4.6: The performance of the centralized learning MAG-DQN compared with decentralized learning MAG-DQN with one to ten agents. The distance travelled are averaged over the last 100 episode among the 800 episodes training on the maps.

4.5.4 Knowledge Adaptation

In the experiments, we found that the knowledge learned by the system maintains a good level of adaptation ability across a different number of agents. This means if we train the model based on a small number of agents, the knowledge obtained by this training can be directly applied by a larger robot team.

As part of our study, we conduct training of the model with three agents on a map and directly use this knowledge to test on multiple agents. As shown in Figure 4.7, the results indicate that training on a small group of agents can be directly transferred to a large group of agents even without fine-tuning. In addition, even without training and fine-tuning, the robot team with more agents can achieve a better performance than a small group of agents. This is a desirable feature for applications in the real world, especially when we cannot train the model exhaustively with a different number of agents.

In Figure 4.8, we train the model using agent teams consisting of two, three, four and

five agents, and apply the learned knowledge to agent teams with six to ten agents. The results indicate that with more agents in the training process, the generalization ability of the learned knowledge is stronger.

Figure 4.7: Apply knowledge from three agents on systems with more agents.

4.6 Conclusion

In this chapter, we discuss the problem of multi-agent environment exploration through the machine learning approach, and present the MAGDQN method for multi-agent system coordinately learning environment exploration strategies.

MAGDQN is built upon the HAC method, which can accurately segment a map and generate a graph representation of the map. Thus, we transform the problem domain from exploring a grid map to exploring a graph. In the next step, through the learning process, the MAGDQN method can generate a fast, efficient environment exploration for an agent team with a various number of agents.

Figure 4.8: Knowledge adaptation by a large team with six to ten agents. We take the highest performance knowledge from teams with two to five agents, and apply the knowledge to larger teams with six to ten agents.

Currently the multi-agent reinforcement learning method can solve the multi-agent environment exploration problem in small and simple environments, but learning to explore large and complicated environment still remains a difficult challenge. However, by applying MAGDQN, we can effectively get a well-performed multi-agent exploration strategy.

In addition, the knowledge adaptation property of the MAGDQN for different number of agents supports the scalability and adaptability of the system in the real-world applications. The user can therefore, keep a well-trained model and apply it on various teams of agents.

Chapter 5

Conclusion & Discussion

5.1 Summary of The Work Done

In this report, we focus on solving the multi-agent environment exploration problem. In the first chapter, we introduce the background of the multi-agent environment exploration, the objective and problem statement of this report, as well as our proposed methods. Chapter 2 discusses the related works to this topic. Chapter 3 and Chapter 4 introduces our proposed methods including the HAC for map segmentation, HAC-based multi-agent environment exploration as well as the MAGDQN method as a multi-agent reinforcement learning system.

The objective of this report is to explore efficient, robust and accurate environment exploration strategies for the multi-agent system. Normally, through considering a partition of the map and allocate agents to explore different areas of the map, the agents can achieve more efficient collaboration in environment exploration. Thus, based on the concept of segmentation, we develop an efficient, fast and accurate map segmentation method (HAC), and furthermore, based on the idea of segmentation, we develop HAC-based multi-agent map exploration method under both partially known and unknown environment.

The main contribution of the HAC method is that we developed the incremental segmentation method based on the ART architecture and apply it for not only segmentation,

but also multi-agent exploration. The incremental segmentation is corresponding with the agent’s capabilities, e.g. the sensor’s detection range, such that the agent is capable to explore the entire segment when it moves to the segment center. It is an advantage of our method comparing to other segmentation based exploration methods, as other segmentation method do not consider the agent’s capability during the segmentation. For example, Voronoi segmentation method tends to cluster an entire large room into one segment even it is difficult for a single agent to explore; our method can clusters the large room into small segments and assign multiple agents to explore that area, which achieves more efficient allocation strategy than the Voronoi segmentation based method.

Another contribution is that we formulate the multi-agent environment exploration problem into a multi-agent graph exploration problem and provide reinforcement learning solutions on this problem. Since reinforcement learning based on the raw map is time-consuming and unstable, we focus on multi-agent learning on a more enriched representation of the map, which is the graph representation, derived from the HAC method. The MAGDQN method provides a graph-based learning mechanism for multi-agent systems to learn a more efficient and better-performed exploration strategy compared to non-learning methods. In addition, through the learning process, the multi-agent exploration is able to outperform the non-learning state-of-the-art methods.

In conclusion, as an underlying method for multi-agent systems in various robotics applications, the multi-agent environment exploration method should be further developed in order to achieve better efficiency, accuracy and robustness. Although many solutions have been developed for autonomously exploring the environment by the multi-agent system including HAC-based method and MAGDQN, there are still large spaces for developing better algorithms and strategies.

5.2 Future Work

As mentioned previously, one of the key points for efficient multi-agent exploration is that each agent is assigned to explore different regions, instead of exploring the same region together. Therefore, based on the concept of segmentation, the HAC-based map exploration method can achieve efficient environment exploration comparing to the state-of-the-art methods. Meanwhile, the experiments for multi-agent map exploration under both partially known environment and unknown environment show the HAC-based exploration achieves better performance than the state-of-the-art methods. However, testing our proposed method under noisy and complicated simulated environment and the real-world environment remains to need to be conducted in order to test the robustness of our methods under different situations. It is a very important performance indicator of the multi-agent exploration methods, as the methods can be used by any kinds of autonomous agents under various situations.

In addition, as discussed in Chapter 4, the deep reinforcement learning for multi-agent map exploration remains a difficult task. Therefore, based on the map segmentation, we present a novel multi-agent graph based deep Q-learning (MAGDQN) to provide better solutions for multi-agent map exploration. Future work can be conducted in several aspects. One of the aspects is to improve the MAGDQN to combine it with the incremental segmentation technique and conduct the learning on a growing graph. Another promising aspect is to develop multi-agent reinforcement learning based on the raw map.

Author's list of publication

- **T. Luo**, B. Subagdja, D. Wang, A. Tan, "Multi-Agent Collaborative Exploration through Graph-based Deep Reinforcement Learning", *2019 IEEE International Conference on Agents (ICA)*, Shan Dong, China, October 18-21, 2019. (Accepted as a full paper and win the best paper award)
- **T. Luo**, Z. Chen, B. Subagdja, A. Tan, "Real-time Hierarchical Map Segmentation for Coordinating Multi-Robot Exploration", *IEEE Access: Regular Papers*. (Under revision)

References

- [1] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, 22(10):1533–1545, 2014.
- [2] F. Amigoni, J. Banfi, and N. Basilico. Multirobot exploration of communication-restricted environments: a survey. *IEEE Intelligent Systems*, 32(6):48–57, 2017.
- [3] T. Andre and C. Bettstetter. Collaboration in multi-robot exploration: to meet or not to meet? *Journal of Intelligent & Robotic Systems*, 82(2):325–337, 2016.
- [4] D. S. Apostolopoulos, L. Pedersen, B. N. Shamah, K. Shillcutt, M. D. Wagner, and W. L. Whittaker. Robotic antarctic meteorite search: Outcomes. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 4174–4179. IEEE, 2001.
- [5] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086*, 2016.
- [6] A. Bautin, O. Simonin, and F. Charpillet. Minpos: A novel frontier allocation algorithm for multi-robot exploration. In *International conference on intelligent robotics and applications*, pages 496–508. Springer, 2012.
- [7] M. Beetz, M. Tenorth, D. Jain, and J. Bandouch. Towards automated models of activities of daily life. *Technology and disability*, 22(1, 2):27–40, 2010.
- [8] R. Bormann, J. Hampp, and M. Hägele. New brooms sweep clean-an autonomous robotic cleaning assistant for professional office cleaning. In *Proceedings of 2015*

REFERENCES

- IEEE International Conference on Robotics and Automation (ICRA 2015)*, pages 4470–4477, 2015.
- [9] R. Bormann, F. Jordan, W. Li, J. Hampp, and M. Hägele. Room segmentation: Survey, implementation, and analysis. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1019–1026, 2016.
- [10] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.
- [11] E. Brunskill, T. Kollar, and N. Roy. Topological mapping using spectral clustering and classification. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 3491–3496. IEEE, 2007.
- [12] K. Bryan and T. Leise. The \$25,000,000,000 eigenvector: The linear algebra behind google. *SIAM Rev.*, 48(3):569–581, Mar. 2006. ISSN 0036-1445. doi: 10.1137/050623280. URL <http://dx.doi.org/10.1137/050623280>.
- [13] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider. Coordinated multi-robot exploration. *IEEE Transactions on robotics*, 21(3):376–386, 2005.
- [14] G. A. Carpenter and S. Grossberg. Adaptive resonance theory. In C. Sammut and G. I. Webb, editors, *Encyclopedia of Machine Learning and Data Mining*, pages 1–17. Springer, Boston, 2016.
- [15] F. R. Chung and F. C. Graham. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.
- [16] R. G. Colares and L. Chaimowicz. The next frontier: combining information gain and distance cost for decentralized multi-robot exploration. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 268–274. ACM, 2016.
- [17] H. Dai, E. B. Khalil, Y. Zhang, B. Dilkina, and L. Song. Learning combinatorial optimization algorithms over graphs. *arXiv preprint arXiv:1704.01665*, 2017.

REFERENCES

- [18] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.
- [19] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai. Deep direct reinforcement learning for financial signal representation and trading. *IEEE transactions on neural networks and learning systems*, 28(3):653–664, 2017.
- [20] S. Dodge, J. Xu, and B. Stenger. Parsing floor plan images. In *Proceedings of the Fifteenth IAPR International Conference on Machine Vision Applications (MVA 2017)*, pages 358–361, 2017.
- [21] H. Endres, W. Feiten, and G. Lawitzky. Field test of a navigation system: Autonomous cleaning in supermarkets. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 2, pages 1779–1781. IEEE, 1998.
- [22] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2137–2145, 2016.
- [23] T. Frank, K.-F. Kraiss, and T. Kuhlen. Comparative analysis of fuzzy ART and ART-2A network clustering performance. *IEEE Transactions on Neural Networks*, 9(3):544–559, 1998.
- [24] J. Friedman, T. Hastie, R. Tibshirani, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000.
- [25] M. Gao and J. M. Zöllner. Local contextual trajectory estimation with demonstration for assisting mobile robot teleoperation. In *Proceedings of 2017 European Conference on Mobile Robots (ECMR 2017)*, pages 1–8. IEEE, 2017.
- [26] B. P. Gerkey and M. J. Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–954, 2004.

REFERENCES

- [27] V. Govindarajan, S. Bhattacharya, and V. Kumar. Human-robot collaborative topological exploration for search and rescue applications. In *Distributed Autonomous Robotic Systems*, pages 17–32. Springer, 2016.
- [28] S. Grossberg. Adaptive pattern classification and universal recoding: I. parallel development and coding of neural feature detectors. *Biological cybernetics*, 23(3): 121–134, 1976.
- [29] S. Gu, E. Holly, T. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3389–3396. IEEE, 2017.
- [30] P.E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.
- [31] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [32] M. Henaff, J. Bruna, and Y. LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.
- [33] D. F. Hougen, S. Benjaafar, J. C. Bonney, J. R. Budenske, M. Dvorak, M. Gini, H. French, D. G. Krantz, P. Y. Li, F. Malver, et al. A miniature robotic system for reconnaissance and surveillance. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 1, pages 501–507. IEEE, 2000.
- [34] A. Howard and N. Roy. The robotics data set repository (radish), 2003. URL <http://radish.sourceforge.net/>.
- [35] Y. Huang, Z. Cao, S. Oh, E. Kattan, and E. Hall. Automatic operation for a robot lawn mower. In *Mobile Robots I*, volume 727, pages 344–355. International Society for Optics and Photonics, 1987.

REFERENCES

- [36] M. Jager and B. Nebel. Dynamic decentralized area partitioning for cooperating cleaning robots. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 4, pages 3577–3582. IEEE, 2002.
- [37] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
- [38] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.
- [39] R. Li, S. Wang, F. Zhu, and J. Huang. Adaptive graph convolutional neural networks. *arXiv preprint arXiv:1801.03226*, 2018.
- [40] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- [41] J. J. Lopez-Perez, U. H. Hernandez-Belmonte, J.-P. Ramirez-Paredes, M. A. Contreras-Cruz, and V. Ayala-Ramirez. Distributed multirobot exploration based on scene partitioning and frontier selection. *Mathematical Problems in Engineering*, 2018, 2018.
- [42] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, pages 6379–6390, 2017.
- [43] N. Mavridis. A review of verbal and non-verbal human–robot interactive communication. *Robotics and Autonomous Systems*, 63:22–35, 2015.
- [44] F. Meyer and S. Beucher. Morphological segmentation. *Journal of visual communication and image representation*, 1(1):21–46, 1990.
- [45] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

REFERENCES

- [46] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.
- [47] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.
- [48] R. R. Murphy. Human-robot interaction in rescue robotics. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 34(2):138–153, 2004.
- [49] A. Nair, P. Srinivasan, S. Blackwell, C. Alcicek, R. Fearon, A. De Maria, V. Panneershelvam, M. Suleyman, C. Beattie, S. Petersen, et al. Massively parallel methods for deep reinforcement learning. *arXiv preprint arXiv:1507.04296*, 2015.
- [50] Óscar Martínez Mozos, R. Triebel, P. Jensfelt, A. Rottmann, and W. Burgard. Supervised semantic labeling of places using information extracted from sensor data. *Robotics and Autonomous Systems*, 55(5), 2007.
- [51] G. Palmer, K. Tuyls, D. Bloembergen, and R. Savani. Lenient multi-agent deep reinforcement learning. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 443–451. International Foundation for Autonomous Agents and Multiagent Systems, 2018.
- [52] C. Sampedro, A. Rodriguez-Ramos, H. Bavle, A. Carrio, P. de la Puente, and P. Campoy. A fully-autonomous aerial robot for search and rescue applications in indoor environments using learning-based techniques. *Journal of Intelligent & Robotic Systems*, 2018. doi: <https://doi.org/10.1007/s10846-018-0898-1>.
- [53] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [54] J. Shi and C. Tomasi. Good features to track. In *Proceedings of 1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1994)*, pages 593–600. IEEE, 1994.

REFERENCES

- [55] M. Simoncelli, G. Zunino, H. I. Christensen, and K. Lange. Autonomous pool cleaning: Self localization and autonomous navigation for cleaning. *Autonomous Robots*, 9(3):261–270, 2000.
- [56] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PloS one*, 12(4):e0172395, 2017.
- [57] S. Thrun, D. Hahnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker. A system for volumetric robotic mapping of abandoned mines. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 3, pages 4270–4275. IEEE, 2003.
- [58] C. Tomasi and T. Kanade. Detection and tracking of point features. *International Journal of Computer Vision*, 9(3), 1991.
- [59] G. Tuna, V. C. Gungor, and K. Gulez. An autonomous wireless sensor network deployment system using mobile robots for human existence detection in case of disasters. *Ad Hoc Networks*, 13:54–68, 2014.
- [60] H. H. Viet, V.-H. Dang, M. N. U. Laskar, and T. Chung. Ba*: an online complete coverage algorithm for cleaning robots. *Applied intelligence*, 39(2):217–235, 2013.
- [61] K. M. Wurm, C. Stachniss, and W. Burgard. Coordinated multi-robot exploration using a segmentation of the environment. In *Proceedings of 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008)*, pages 1160–1165. IEEE, 2008.
- [62] B. Yamauchi. Frontier-based exploration using multiple robots. In *Proceedings of the second international conference on Autonomous agents*, pages 47–53. ACM, 1998.
- [63] B. Yamauchi. Autonomous urban reconnaissance using man-portable ugvs. In *Unmanned Systems Technology VIII*, volume 6230, page 62300S. International Society for Optics and Photonics, 2006.

- [64] H. Zender, O. M. Mozos, P. Jensfelt, G.-J. Kruijff, and W. Burgard. Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems*, 56(6):493–502, 2008.
- [65] D. Zhu, T. Li, D. Ho, C. Wang, and M. Q.-H. Meng. Deep reinforcement learning supervised autonomous exploration in office environments. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7548–7555. IEEE, 2018.
- [66] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3357–3364. IEEE, 2017.
- [67] Z. Zivkovic, B. Bakker, and B. Krose. Hierarchical map building and planning based on graph partitioning. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 803–809. IEEE, 2006.