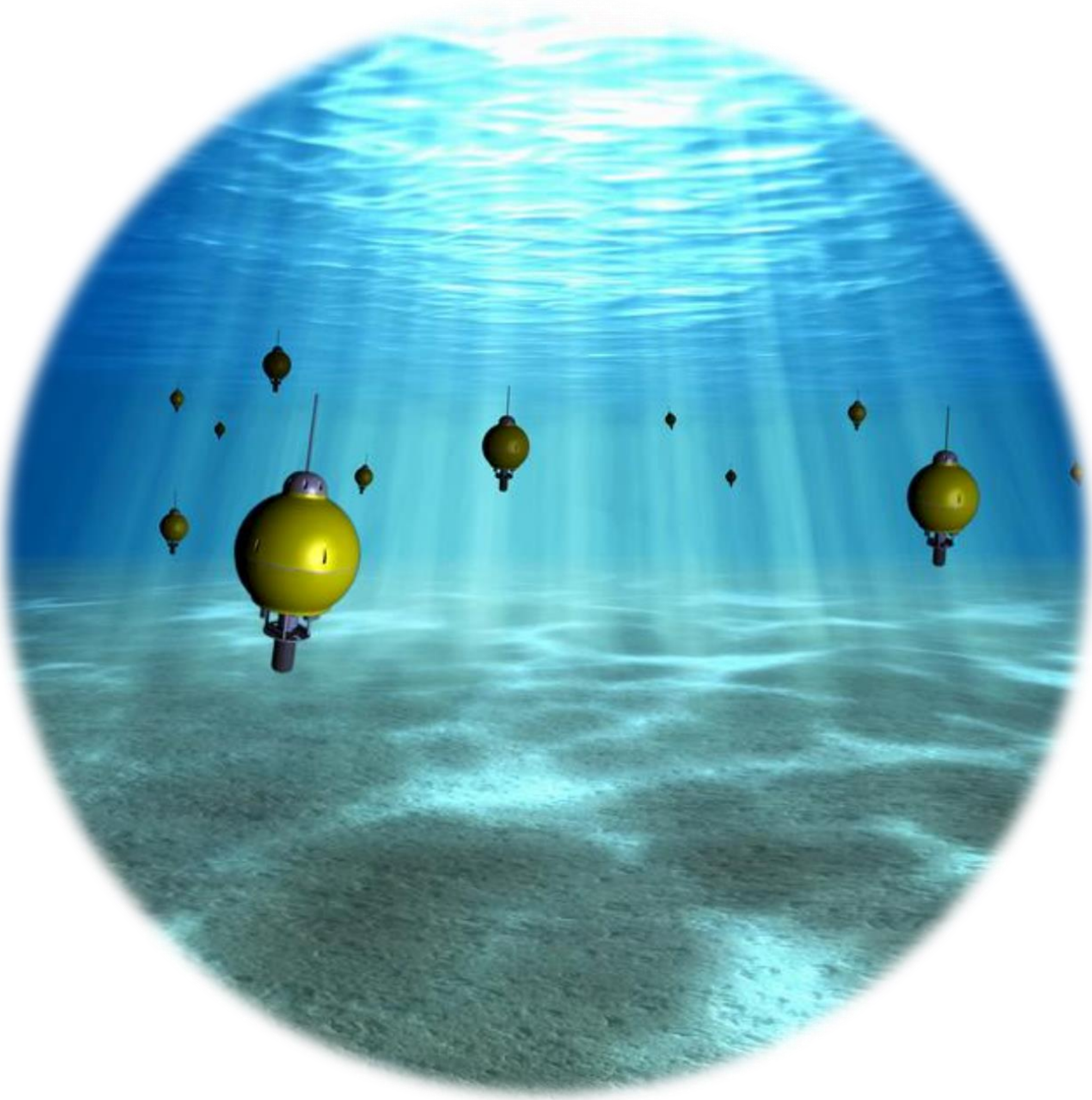
Internship Report on Autonomous Underwater Vehicles

KASSAMALY Mourtaza
mourtaza.kassamaly@ensta-bretagne.org

ENSTA Bretagne, 2 rue F. Verny, 29806 Brest Cedex 9, France



Keywords

Underwater Sensor Networks, Autonomous Underwater Vehicles, Monte-Carlo Tree Search, Acoustic, Data Ferrying, Waypoints, Shortest Path, Dynamic Multiple Traveling Salesman Problem.

Abstract

Underwater Sensor Networks are used to monitor oceans which is an environment that restrains communication. Forwarding data from deep sensor nodes to the surface becomes a challenge that can be tackled by using multiple Autonomous Underwater Vehicles. The goal is to find the shortest path to come close enough to each sensor node and to smartly split the workload among all available vehicles in an environment including static and mobile obstacles. A decentralized version of the Monte-Carlo Tree Search algorithm is experimented in this paper and shows some good results in tackling similar and simplified problems.

Mots-clés

Réseaux de capteurs sous-marins, Véhicules sous-marins autonomes, Recherche par arborescence de Monte-Carlo, Acoustique, Transfert de données, Points d’intérêts, Le plus court chemin, Problème des multiples voyageurs de commerce dynamique.

Résumé

Les réseaux de capteurs sous-marins sont utilisés pour surveiller les océans, un environnement qui restreint la communication. La transmission des données des nœuds de capteurs en profondeur vers la surface devient un défi qui peut être relevé en utilisant plusieurs véhicules sous-marins autonomes. L'objectif est de trouver le chemin le plus court pour s'approcher suffisamment de chaque nœud de capteur et de répartir intelligemment la charge de travail entre tous les véhicules disponibles dans un environnement comprenant des obstacles statiques et mobiles. Une version décentralisée de l'algorithme de recherche par arborescence de Monte-Carlo est expérimentée dans cet article et montre quelques bons résultats dans la résolution de problèmes similaires et simplifiés.

Content

Keywords 2
Abstract 2
Mots-clés 2
Résumé 2
1. Internship setting 4
1.1. Host Laboratory 4
1.2. Context 4
2. Introduction 6
3. Algorithms for Decentralized AUVs Data Ferrying with Dynamic Obstructions 7
3.1. Monte Carlo Tree Search 7
3.2. Problem Definition 9
3.3. Finding waypoints with Quantum-behaved Particle Swarm Optimization 10
3.4. Finding a path in an uncertain environment with a Decentralized-MCTS algorithm 11
3.5. Evaluation Results 12
4. Conclusion 13
Glossary 15
References 16
Appendices 17

1. Internship setting

1.1. Host Laboratory

The internship was supported by Thales South Australia and hosted by the Centre Defence Communication and Networking laboratory in the University of Adelaide, Adelaide, Australia. Its main activity is to model and simulate communication bearers, it conducts research for the nation's defence sector on advanced defence communication, information networking engineering and system design and expertise.

1.2. Context

In order to monitor oceans, several sensors can be deployed at different locations and different depths. The deployment of those sensors forms a network composed of several nodes (sensor nodes) which is called an Underwater Sensor Network (USN) **[1]**. Sensor nodes can communicate, so that the collected data can be brought back to the surface or to a ground station in order to be processed. Since electromagnetic and radio waves are not adapted for communicating underwater, the preferred mean is to use acoustic waves that allows a high range with low frequencies. It is also known as Sound Navigation and Ranging (SONAR). As **[1]** describes, underwater acoustic communication is tricky. Because of the fluctuation in temperature, in salinity and the change in pressure, the sound speed varies with depth. Consequently, acoustic waves do not travel straight and can reflect on the bottom of the ocean or on the surface, increasing the information delay. The bandwidth is also limited, and the error probability is not to be ignored.

In order to cope with these constraints, it is common to use wireless multi-hop and ad hoc communication. Such a network is called a Mobile Ad Hoc Network (MANET). This means that data collected by deep sensors are going to be sent back to the surface by forwarding the data little by little by using other sensors closer to the surface as relays. It is important to bring the data to the surface as it can be sent to a ground station to be processed. The reason for that is in case of a sensitive detection by a sensor node, it will not be able to do something about it but rather will only be capable of forwarding the information. To do this, a mobile sensor node can be sensing other sensor nodes so that the information is forwarded rapidly in a Wireless Sensor Network (WSN). Or, it can store the message in its memory until another sensor node is found close enough to forward the data. Consequently, the timeliness of the information is not guaranteed, and such a network is called a Delay Tolerant Network (DTN). Those sensor nodes could be any kind of Underwater Vehicles such as Remote Operated Vehicles (ROV), Unmanned Underwater Vehicles (UUV) or Autonomous Underwater Vehicles (AUV).

The type of communication, the selected routes to bring data to the surface are known as routing protocols. They are used to forward the data and several of them are discussed in **[2]** which is a survey of different routing protocols. They are based on geographical parameters, mobility of sensor nodes, battery levels, networks' density, information sensitivity and sensor node's capability. Given those characteristics and according to a specific application, one routing protocol could appear to be more efficient than others. In most of applications, a multi-hop approach is used. However, a major flaw of this method is discussed in **[3]**, as it is pointed out that sensor nodes closer to the surface are more likely to receive requests from deeper sensor nodes and thus will run out of battery faster. This imbalance will leave some sensor nodes unusable and may compromise an entire mission. It is proposed to use AUVs to deal with this issue and use them to collect non sensitive data from underwater static sensor nodes.

Article **[2]** describe one routing protocol that is particularly interesting when AUV are involved: The Location-based Clustering Algorithm for Data Gathering (LCAD). Considering an area of interest, sensor nodes will be deployed at different locations and different depths such that they can make relevant measurements. We can imagine those sensor nodes will be organized in different clusters based on their locations and each cluster will contain a leader node designated among the several sensor nodes in that cluster. This leader will most likely be in the middle of the cluster and will gather data from the other sensor nodes. Thus, the data does not need to be immediately brought to the surface but only to the cluster leader. Typically, low frequencies acoustic waves could be used for gathering data among each cluster. A group of AUVs will be used as “data mules” to collect the gathered data from each cluster leaders. Typically, high frequency acoustic waves could be used for transferring data from cluster leaders to AUVs. AUVs will then be acting themselves as mobile underwater sensor nodes which aims at giving support to the rest of the network.

This rises many challenges. The first one is to collect and bring the data fast enough, so that the data is processed before it gets obsolete. Another one is to optimize the number of AUVs used and their paths to reduce buying costs and energy consumption. And the last one is to make them cooperate in order to complete the first two challenges while coping with the unforeseen (obstacles or adversaries) which is especially challenging with restricted communication possibilities.

2. Introduction

To remain competitive, this must be done by optimizing the trajectory as the set of sensor nodes will be dealt with using multiple AUVs moving around as a swarm. The first question one can ask is about sensor nodes moving below the surface. Underwater currents can make sensor nodes move from their initial position, thus making it necessary to regularly recompute each node’s location. This could result in an energy loss.

Hypothesis 1. Sensor nodes are supposed anchored to the bottom or to a floating buoy such that their position remains the same during the whole mission. We can suppose that they are evenly distributed in an area of interest to monitor mines without leaving any dead angle. The range of the sensors must be considered.

Another question is how to optimize the trajectory of an AUV which must reach some cluster leaders to bring back data to the surface. This is to say, is the priority given to the trajectory that will take the least amount of time or the least amount of energy ?

Hypothesis 2. According to **[4]**: “The vehicle’s water-referenced velocity is assumed to be constant. Since this velocity is proportional to the cube root of the thrust, equivalently, the vehicle has constant thrust power and thus the energy consumed for a path is a constant multiple of the distance travelled.”. This hypothesis ensures that optimizing the travel time, the travelled distance and the energy consumption are equivalent.

Hypothesis 3. AUVs are supposed to have enough power to perform the mission by following the path minimizing the travelled distance. The amount energy needed can still be computed later by considering the energy needed to travel from one sensor node to another and to collect data from each sensor node.

This problem differs from a classical swarm robotics issue since in this case, the goal is not to move a swarm of robots as a group with one leader but rather to split the available space between several AUVs so that the area of interest is covered in the minimum amount of time and so that each AUV travels the least possible distance.

3. Algorithms for Decentralized AUVs Data Ferrying with Dynamic Obstructions

3.1. Monte Carlo Tree Search

The Monte Carlo Tree Search is an algorithm that seeks to find the most rewarding move regarding a given goal at a given time and a given state in a mission or game. It is based on a search tree with different arborescence that will grow and evolve as it is used. The MCTS algorithm is similar to the minimax algorithm but is less time consuming. The minimax algorithm considers, at each state, all the different states that can be reached from the current state. As a result, absolutely all scenarios are considered, and the algorithm is sure to find a suitable action. However, projecting dozens of steps into the future while considering all possible scenarios at each step can quickly increase the memory used and the time needed to converge. The MCTS algorithm solves this problem by not considering all the possible scenarios at each step but only a few: first by chance, then by preferentially selecting the scenarios that have given a high reward, and sometimes by preferring scenarios still unexplored. It balances exploitation and exploration and covers the most promising scenarios, thus saving IT resources.

The tree is composed of leaves and branches, or nodes and arches. Each leaf or node represents a game state and each branch or arc represents a transition between two game states by an action of one of the players. The algorithm is used in two parts: training and exploiting.

The first part, training, is divided into 4 steps. The completion of all the 4 steps is often called a rollout. Each rollout will update the search tree and one can do as much rollout as desired. The number of rollouts will surely depend on how much time do we have before making a decision in the game.

1. The selection consists in selecting an unvisited node. The selection is done according to rules or a policy to be defined, random being one of the possibilities.
2. The expansion consists in choosing a child node from the newly selected node, that is to say that the execution of an action in accordance with the rules of the game allows to obtain this child node from the node selected in step 1.
3. The simulation consists in executing a random game from the extended node found in step 2, (this could be done randomly or according to a policy to be defined) until reaching a final node (until the game reaches a final state).
4. Back propagation consists in back propagating the reward obtained at the end of the simulation through the nodes encountered. Each node then contains two pieces of information that are kept up to date: the average reward obtained starting from this node, and the number of times this node has been visited by the algorithm.

The second part is the exploiting part, it consists in using the updated tree from the training to make a decision in the game. Considering the state of the game we are in (the node), we have to make a decision (make a move in the game according to the rules), which is to say select which child node to choose according to the updated tree. The node is chosen by maximizing

$$a = r + c * \sqrt{\left(\ln \frac{(N)}{n}\right)}$$

where r is the average reward among all simulations obtained starting from this node, N is the number of times the parent node has been visited, n the number of times the considered node has been visited and c is an exploration parameter which gives weight to unvisited nodes to encourage exploration. This last coefficient is meant to be chosen empirically.

In brief, the Monte Carlo Tree Search is a method that uses a search tree that allows to forecast the most likely scenarios from a given state and use it to find the most promising moves towards a given objective.

3.2. Problem Definition

The area of interest is delimited by a 50 kms x 50 kms rectangle and a maximum depth of 30 meters (typically a port exit). This area is evenly filled with sensor nodes anchored to the bottom in such a way that the whole area can be monitored without leaving any dead angles. The set of N sensors is **S** = {s1, s2, ..., sN}. All the sensors have the same communication radius known as R. Within a circle of radius R around each sensor node, it is considered that the communication quality is high, and that the sensor node can properly communicate data using acoustic waves within the range without losing information. The area is meant to be covered by a fleet of m AUVs. Altogether, they will visit a set of M waypoints that will make AUVs come close enough to each sensor nodes in order to enable communication and therefore forward data. The set of waypoints is **WP** = {w1, w2, ..., wM}. The higher the number of sensor nodes covered by each waypoint, the better it is. The "waypoints" issue is presented as follows:

minimize *M* the total number of needed waypoints

s.t

$$\forall i \in \{1 \dots M\}\ \exists\, j\ \in\ \{1 \dots N\}\ such\ that\ |w_i - s_j| \leq R$$

(each waypoint covers at least one sensor node)

$$\forall j \in \{1 \dots N\}\ \exists\, i\ \in\ \{1 \dots M\}\ such\ that\ |w_i - s_j| \leq R$$

(every sensor node is covered)

Once the waypoints are determined. The "path-planning" issue aims at finding a path for each AUV to follow so that every waypoint is visited with the minimum overall cost. A given number of AUVs will start at the same location defined as the origin and each one of them will be following a different path covering some waypoint and such that each waypoint is visited at least once by an AUV. This problem is similar to a multiple Traveling Salesman Problem except that obstacles are considered in the space, some of them are known and static, others are unforeseen and mobile and act as enemies that AUVs will have to avoid. The mission is completed by using m AUVs. Let us note Xk(t) = (xk(t), yk(t), zk(t)) the trajectory of the k-th AUV. **SO** is a set of static obstacles and **MO(t)** is a set of mobile obstacles. The AUVs must altogether visit all the waypoints while avoiding all obstacles. The mission starts at *t = t0* and ends at *t = tf*. The "path-planning" issue is presented as follows:

$$\forall k \in \{1 \dots M\}\, X_k(t_0) = X_k(t_f) = (0\ 0\ 0)$$

(all AUVs start at the same starting location)

$$\forall i \in \{1 \dots M\}\ \exists\, t \in [t_0; t_f]\ and\ k \in \{1 \dots M\}\ such\ that\ X_k(t) = w_i$$

(all waypoints are covered at least once)

$$\forall t \in [t_0; t_f]\ \forall k \in \{1 \dots M\}\ X_k(t) \notin \mathrm{SO(t)} \cap \mathrm{MO(t)}$$

(AUVs must avoid all obstacles)

3.3. Finding waypoints with Quantum-behaved Particle Swarm Optimization

We will build on the study conducted in **[6]** which aims at planning a tour for an AUV to follow in order to visit all the sensor nodes with the minimum distance travelled and for the sensor nodes to reduce their energy consumption. In their studies, the researchers propose to find a shortest path traveling several waypoints placed close enough to sensor nodes. Since sensor nodes have a communication range R, the idea is to find waypoints that would cover several sensor nodes at a time so that an AUV would not have to travel to all sensor nodes but rather travel to those waypoints. Sensors' positions and ranges are used with an algorithm based on a Quantum-behaved Particle Swarm Optimization (QPSO) to find waypoints for the AUV to visit. They suppose that all the sensor nodes are on a horizontal plane, which means that they are at the same depth and that all sensor nodes can have a random range. In our case, we will consider the problem in 3D meaning that the depth can vary among the sensor nodes, and that the sensor nodes are of the same type meaning that they have the same communication range. In that case the algorithm proposed in the study can be a little adapted.

The symbol **S** = {s1, s2, ..., sN} represents a set of sensors and (s_{ix}, s_{iy}, s_{iz}) the coordinates of the i^{th} sensor. All the sensors have the same communication radius known as *R*. First, a QPSO algorithm is used to find waypoints. Those waypoints will have to cover the most sensor nodes as possible and be as close as possible to them in order to reduce energy consumption of both the sensor nodes and the AUV. Let us note that the algorithm described below will only output one waypoint, the one that will cover the most sensor nodes. It will have to be executed multiple times until all sensors are finally covered. As a result, multiple executions of this algorithm will return the minimum waypoints to cover all the sensors **WP** = {w1, w2, ..., wM}. The algorithm is shown in Figure 1 in Appendix A.

3.4. Finding a path in an uncertain environment with a Decentralized-MCTS algorithm

Consider a set of waypoints ***WP*** = {w_1, w_2, ..., w_M} (for instance those found by the QPSO algorithm) and *m* agents (for instance AUVs) that have to reach altogether all the waypoints by minimizing their traveled distance and by avoiding static and mobile obstacles. Static obstacles are simple objects that can be sensed by agents when they come in their neighborhood. Mobile obstacles are actually enemies trying to block the path of agents to disturb their plans. They are equipped with a minimalist smartness which means they move around randomly to disturb agents, but they try not to collide with them as they do not want to be damaged themselves. Constraints are meant to be added progressively in order to get closer to the reality as the solution matures. We consider going from 2D to 3D, adding communication constraints, ocean currents. At first, we consider a 2D grid of size *L* composed of:

- *n* targets/waypoints randomly placed
- *m* agents at the origin of the grid with the knowledge of targets location
- *k1* static obstacles
- *k2* mobile obstacles/enemies

Agents will be moving from one case to another in the grid while avoiding any kind of obstacles in order to collect targets. Each target is to be collected once and the mission ends when all the waypoints have been reached, meaning that all the targets have been collected and all the available agents are back to the origin of the grid. Each agent will select which waypoint to deal with using a Decentralized-MCTS algorithm building on the study conducted in **[7]**. Each agent has its own MCTS tree and will use a MCTS algorithm to forecast possible events and future actions as explained previously. The MCTS tree will be used to choose the next waypoint to travel to.

Each agent carries a probability distribution over the remaining waypoints and will keep updating it, it also has the knowledge of the probability distribution of other agents. Each agent will be working on its own but is able to communicate its probability distribution to other agents so that each agent has information about the probability distribution of all agents. It will use them to guess what possible next moves other agents could do and act accordingly, this is how we create cooperation between agents. Probability distributions are updated using Algorithm 3 of **[5]** with two particularities: the set of feasible action sequences for each agent $\widehat{\mathrm{X}}^{\mathrm{r}}$ is the set of remaining waypoints (not collected yet) but ordered by the distance to the position of the agent, and a sample $x \in \hat{\mathrm{X}} = \{\widehat{\mathrm{X}}^1, \widehat{\mathrm{X}}^2, \ldots, \widehat{\mathrm{X}}^{\mathrm{m}}\}$ is a set of actions taken from each of the feasible action sequences over all agents. We choose to plan one step ahead, meaning that only one waypoint is picked from each of the feasible action sequences for each agent.

Agents will reset their MCTS trees and restart the algorithm with a new configuration in the following situations: at the beginning of the mission, when any agent reaches a target and when any agent encounters an obstacle when moving toward its assigned target. When performing the rollout, the agent will forecast what other agents could do using their respective probability distributions then choose a waypoint for itself by selecting the nearest non-selected waypoint. The pseudo code of the mission is shown in Figure 2 in Appendix B.

3.5. Evaluation Results

For conducting tests, 4 agents are placed at the origin of a 10*10 grid with 10 waypoints, 3 static obstacles and 3 mobile obstacles randomly placed. Each time an agent travels from one case to another, it counts as a step and the number of steps of each agent has been recorded during the simulations. The graphs on Figure 3, Figure 4, Figure 5 and Figure 6 in Appendix C show the evolution of the number of steps through several simulations for each agent. Simulations were performed with a Decentralized Multi-Agent MCTS algorithm as described in the paper on one hand. One the other hand, a Classical Multi-Agent MCTS algorithm has been used. It is the exact same as the decentralized algorithm except that probability distributions are not updated and shared between agents. Results show that performance are more or less the same when switching from a classical MCTS algorithm to a decentralized version. The particularity of the decentralized version is the updating of probability distributions. In both versions of the algorithm, each agent tries to sample possible moves from other agents thanks to their respective probability distributions. In the classical version, all probabilities are equal while in the decentralized version, probabilities are updated thus they are unevenly distributed. The "not so good" performance of the decentralized version could be explained by the fact that in the decentralized version, the algorithm updating probability distributions plans one step. We can imagine modifying this algorithm to plan several steps ahead could improve performance. The MCTS algorithm is a state-of-the-art algorithm and it still requires time and work to adapt it to a dynamic data ferrying problem. This work is meant to be continued in the future by other students and will surely lead to some very interesting results in the field.

4. Conclusion

The subject of the internship was chosen by myself with the coordination of my tutor, Mr. Hung NGUYEN who helped me formulating my ideas and redirected me in the right direction when trying to state a problem. He was interested in me trying to define a subject by myself so that there is a good compromise between what he wants and what I want, what I am interested in and what I studied so far. As the internship went forward, the subject was meant to be slightly modified and adapted to match the literature and to have interesting outcomes. The challenge of using AUVs for Data Ferrying satisfies both of us because dealing with AUVs is what I expected and dealing with data forwarding and

USN is what my tutor expected. The results are of the simulations are promising but not sufficient. The work is meant to be continued by other students in the future so that the CDCN can publish a paper out of it.

This internship is the first time I really jumped into some serious research projects and I found it very interesting. I appreciated the fact that I had to look for information and for knowledge that I don't have and I appreciated the fact that I managed to learn a lot of new concepts that I did not know about in a very short time. It was reassuring to realize that our formation is relevant in terms of what happens in the "real-world" (meaning, out of the school). Participating in a research project takes time and one have to deal with different momentum. Sometimes, I knew in which direction I was moving, and the motivation was high. Some other times, the outcome was blurred, and I had to look at the literature for a few days before clearing my mind and thus the motivation was low.

I found that people doing research are really passionate about their field because it requires a lot of work, concentration and perseverance to keep the pace of state-of-the-art concepts, algorithms and keep publishing papers. I cherish this experience as I learned a lot during this internship, but I do know that this is not what I want to do in the future. As a military student, I had the opportunity to experiment being more like a manager in the French Navy and I am more comfortable picturing myself in this type of functions than in a pure scientific function. I like to organize, to have a global and a more general point of view more than I like focusing on a specific concept and problem. This internship was the opportunity to learn scientific knowledge, to discover a research project and participate in it and to find what kind of work I don't see myself doing in my daily life in the future. These observations are to be mitigated since I did not experiment working in a team, in an office with other researchers but rather alone in my bedroom with a tutor that I was only seeing in video twice a week. My point of view and what I felt about the internship would probably change if the situation were different.

Glossary

CDCN: Centre Defence Communication and Networking

USN: Underwater Sensor Network

SONAR : SOund NAvigation and Ranging

AUV : Autonomous Underwater Vehicle

MCTS : Monte-Carlo Tree Search

Dec-MCTS : Decentralized Monte-Carlo Tree Search

References

[1] J. Sánchez-García, J.M. García-Campos, M. Arzamendia, D.G. Reina, S.L. Toral, D. Gregor. A survey on unmanned aerial and aquatic vehicle multi-hop networks: Wireless communications, evaluation tools and applications. Computer Communications 119 (2018) 43–65.

[2] Muhammad Ayaz, Imran Baig, Azween Abdullah, Ibrahima Faye. A survey on routing techniques in underwater wireless sensor networks. Journal of Network and Computer Applications 34 (2011) 1908–1927.

[3] Chien-Fu Chenga, Lung-Hao Li. Data gathering problem with the data importance consideration in Underwater Wireless Sensor Networks. Journal of Network and Computer Applications 78 (2017) 300–312.

[4] Zheng Zeng, Karl Sammut, Lian Lian, Fangpo He, Andrew Lammas, Youhong Tang. A comparison of optimization techniques for AUV path planning in environments with ocean currents. Robotics and Autonomous Systems 82 (2016) 61–72

[5] Bradberry, Jeff (2015-09-07). "Introduction to Monte Carlo Tree Search".

[6] Chen, Mingzhi and Daqi Zhu. “Data collection from underwater acoustic sensor networks based on optimization algorithms.” Computing 102 (2019): 83-104.

[7] G. Best, O. M. Cliff, T. Patten, R. R. Mettu, R. Fitch, Dec-MCTS: Decentralized planning for multi-robot active perception, The International Journal of Robotics Research 38 (2-3) (2019) 316–337. 33

[8] RobAid. “Sensor Networks of Freely Drifting Autonomous Underwater Explorers,” November 14, 2009. http://www.robaid.com/robotics/sensor-networks-of-freely-drifting-autonomous-underwater-explorers.htm.

Appendices

Appendix A: Algorithm 1

Algorithm 1 Find a waypoint with a QPSO algorithm

Input

- A set of N sensors $\boldsymbol{S} = \{s_1, s_2, \ldots, s_N\}$ and a range R

Output

- One waypoint covering as much sensors as possible denoted by G

Initialize

- Initialize N particles by $x_i = s_i \ \forall i = 1, 2, \ldots N$
- Initialize their best locations so far by $P_i = x_i \ \forall i = 1, 2, \ldots N$
- Initialize C their average best location by $\sum_{i=1}^{N} P_i$
- Initialize G the best particle's location by C

Repeat

- Fetch the average best location C by $\sum_{i=1}^{N} Pi$
- **for** each particle x_i **do**
  - Calculate the particle's covering sensors number
  - Record the covering sensors number f_{Pi} and f_{xi} for P_i and x_i
  - Update its best location P_i by

    $$P_i = x_i \ if \ and \ only \ if \ f_{xi} > f_{Pi}$$
  - Record the covering sensors number f_G for G
  - Update the global best particle G by

    $$G = P_i \ if \ and \ only \ if \ f_{Pi} > f_G$$
  - Get a random value for each dimension j of the particle's location by

    $$p_{ix} = \phi_1 \cdot P_{ix} + [1 - \phi_1] \cdot G_x, \ \phi_1 \sim U(0,1)$$
    $$p_{iy} = \phi_2 \cdot P_{iy} + [1 - \phi_2] \cdot G_y, \ \phi_2 \sim U(0,1)$$
    $$p_{iz} = \phi_3 \cdot P_{iz} + [1 - \phi_3] \cdot G_z, \ \phi_3 \sim U(0,1)$$
  - Update each dimension j of the particle's location x_i by

    $$x_{ix} = p_{ix} \pm 0.7 \cdot |C_x - x_{ix}| \cdot ln\left[\frac{1}{u_1}\right], \ u_1 \sim U(0,1)$$
    $$x_{iy} = p_{iy} \pm 0.7 \cdot |C_y - x_{iy}| \cdot ln\left[\frac{1}{u_2}\right], \ u_2 \sim U(0,1)$$
    $$x_{iz} = p_{iz} \pm 0.7 \cdot |C_z - x_{iz}| \cdot ln\left[\frac{1}{u_3}\right], \ u_3 \sim U(0,1)$$
- **end for**

Until the best particle's location G is not changed for more than 100 iterations

Figure 1: 3D QPSO for finding optimal waypoints

Appendix B: Algorithm 2

Algorithm 2 Finding a path with a Decentralized-MCTS algorithm

Input

- Set of waypoints

  (The set of static obstacles and the set of mobile obstacles are unknown)

Initialize

- Initialize *m* agents at the origin of the space
- **for** each agent **do**
  - Initialize probability distribution
  - Communicate position and probability distributions to other agents
  - Initialize MCTS tree
  - Sample possible moves for other agents using probability distributions
  - Train the MCTS tree with rollouts (selection, expansion, simulation, back-propagation)
  - Use the MCTS tree assign a non-chosen waypoint to the agent
- **end for**

Repeat

- **for** agent **do**
  - Sense the surrounding environment
  - Make a move toward the assigned waypoint if possible
  - Update probability distributions with Algorithm 3 of **[5]**
  - **if** any waypoint has been reached by the agent
    - For each agent, redistribute the probability of choosing this waypoint over the remaining waypoints
  - **end if**
- **end for**
- **if** a target has been reached during this iteration <u>OR</u> if any agent has been blocked in his path by an obstacle
  - Reset all MCTS trees but keep probability distributions
  - Perform rollouts and choose a waypoint for every agent
- **end if**

Until all waypoints are visited and all agents are back to the origin of the space

Figure 2: Decentralized-MCTS Algorithm

Appendix C: Evaluation results

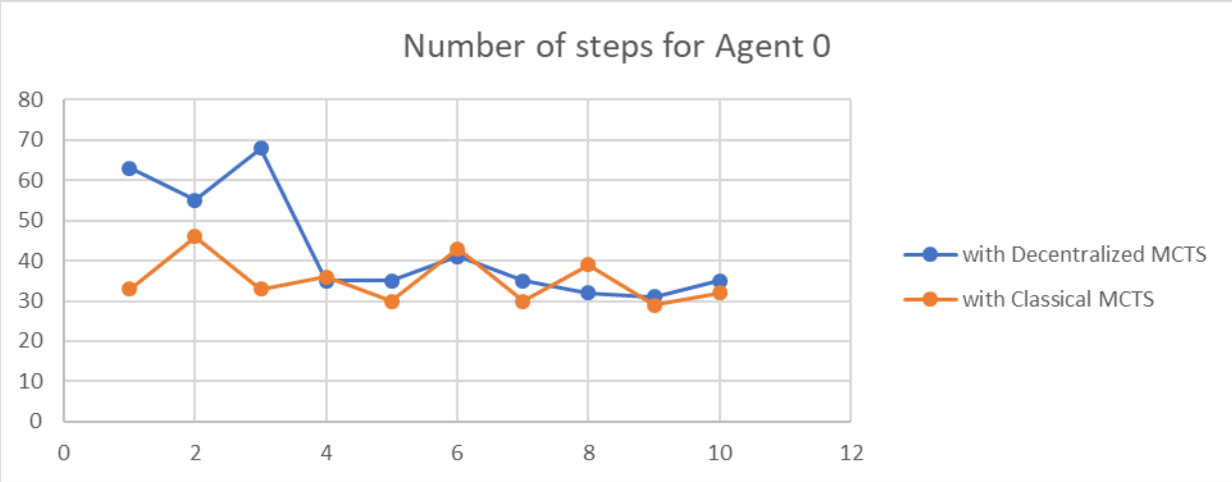


Figure 3: Number of steps for agent 0

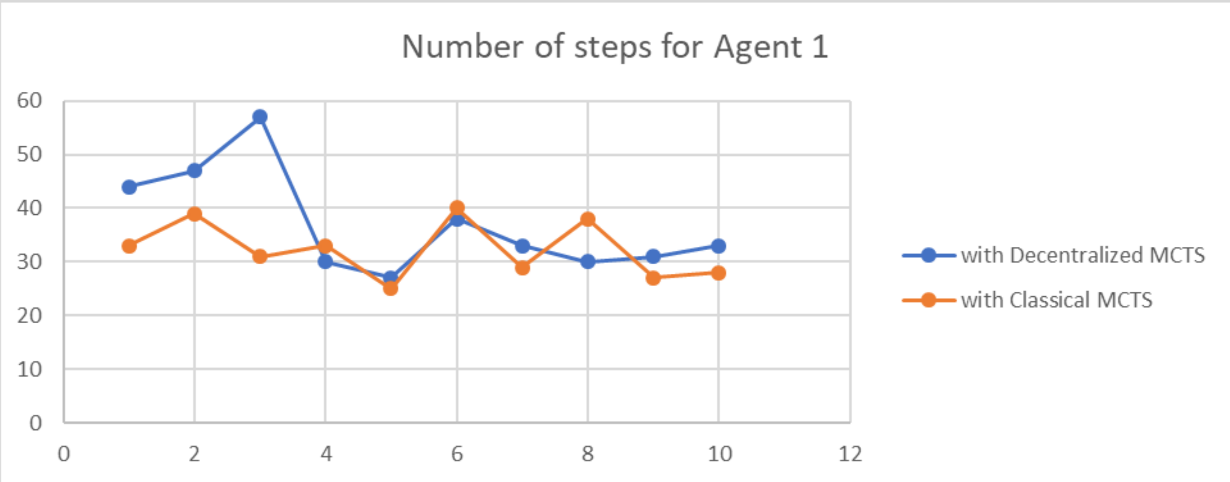


Figure 4: Number of steps for agent 1

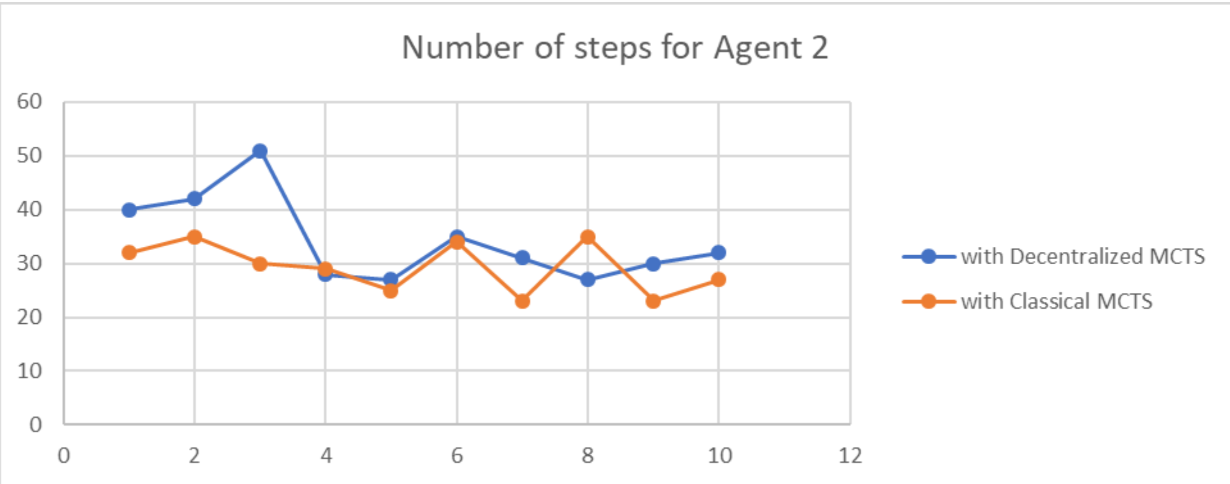


Figure 5: Number of steps for agent 2

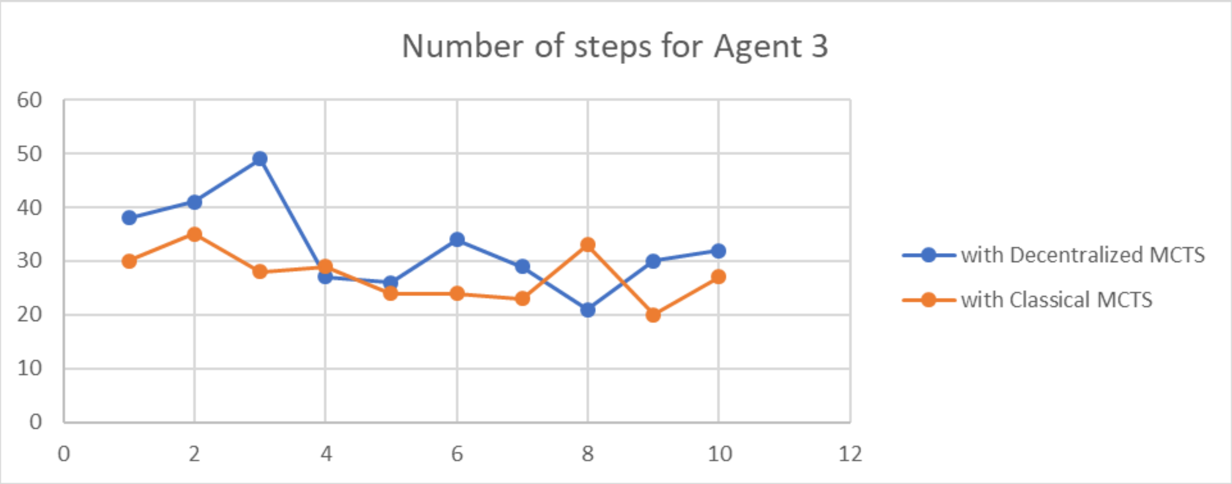


Figure 6: Number of steps for agent 3

Appendix D: Internship assessment report

ENSTA Bretagne

RAPPORT D'EVALUATION
ASSESSMENT REPORT

Merci de retourner ce rapport par courrier ou par voie électronique en fin du stage à :
At the end of the internship, please return this report via mail or email to:

ENSTA Bretagne – *Bureau des stages - 2 rue François Verny - 29806 BREST cedex 9 – FRANCE*
☎ 00.33 (0) 2.98.34.87.70 / stages@ensta-bretagne.fr

I - ORGANISME / *HOST ORGANISATION*

NOM / *Name* The University of Adelaide

Adresse / *Address* Adelaide SA 5005 Australia

Tél / *Phone (including country and area code)* +61 8 8313 4455

Nom du superviseur / *Name of internship supervisor* Hung Nguyen

Fonction / *Function* Senior Research Fellow

Adresse e-mail / *E-mail address* hung.nguyen@adelaide.edu.au

Nom du stagiaire accueilli / *Name of intern* Mourtaza KASSAMALY

II - EVALUATION / *ASSESSMENT*

Veuillez attribuer une note, en encerclant la lettre appropriée, pour chacune des caractéristiques suivantes. Cette note devra se situer entre **A (très bien)** et **F (très faible)**
Please attribute a mark from ***A (excellent)*** *to* ***F (very weak).***

MISSION / *TASK*

- La mission de départ a-t-elle été remplie ? *Was the initial contract carried out to your satisfaction?* — (A) B C D E F

- Manquait-il au stagiaire des connaissances ? *Was the intern lacking skills?* — ☐ oui/*yes* ☒ non/*no*

  Si oui, lesquelles ? / *If so, which skills?* ______

ESPRIT D'EQUIPE / *TEAM SPIRIT*

- Le stagiaire s'est-il bien intégré dans l'organisme d'accueil (disponible, sérieux, s'est adapté au travail en groupe) / *Did the intern easily integrate the host organisation? (flexible, conscientious, adapted to team work)* — A (B) C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* It's a remote internship so we could not fully evaluate this category.

7

Version du 05/04/2019

COMPORTEMENT AU TRAVAIL / *BEHAVIOUR TOWARDS WORK*

Le comportement du stagiaire était-il conforme à vos attentes (Ponctuel, ordonné, respectueux, soucieux de participer et d'acquérir de nouvelles connaissances) ?
Did the intern live up to expectations? (Punctual, methodical, responsive to management instructions, attentive to quality, concerned with acquiring new skills)?

A B C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* ____

INITIATIVE – AUTONOMIE / *INITIATIVE – AUTONOMY*

Le stagiaire s'est –il rapidement adapté à de nouvelles situations ? A B C D E F
(Proposition de solutions aux problèmes rencontrés, autonomie dans le travail, etc.)

Did the intern adapt well to new situations? A B C D E F
(eg. suggested solutions to problems encountered, demonstrated autonomy in his/her job, etc.)

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* ____

CULTUREL – COMMUNICATION / *CULTURAL – COMMUNICATION*

Le stagiaire était-il ouvert, d'une manière générale, à la communication ? A B C D E F
Was the intern open to listening and expressing himself /herself?

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* ____

OPINION GLOBALE / *OVERALL ASSESSMENT*

- La valeur technique du stagiaire était : A B C D E F
  Please evaluate the technical skills of the intern:

III - PARTENARIAT FUTUR / *FUTURE PARTNERSHIP*

- Etes-vous prêt à accueillir un autre stagiaire l'an prochain ?
  Would you be willing to host another intern next year? ☒ oui/*yes* ☐ non/*no*

Fait à ____, le ____
In February, on 2021

Signature Entreprise ____ Signature stagiaire
Company stamp ____ *Intern's signature*

Merci pour votre coopération
We thank you very much for your cooperation

8

Version du 05/04/2019