GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

ISSN 1612-6793
Nr. ZFI-BM-201x-xx

Master's Thesis
submitted in partial fulfilment of the
requirements for the course "Applied Computer Science"

Topology Control Using Fuzzy Game Theory in Mobile Underwater Sensor Networks

Chencheng Liang

Institute of Computer Science

Master's Theses
of the Center for Computational Sciences
at the Georg-August-Universität Göttingen

05. March 2018

Georg-August-Universität Göttingen
Institute of Computer Science

Goldschmidtstraße 7
37077 Göttingen
Germany

☎ +49 (551) 39-172000
FAX +49 (551) 39-14403
✉ office@informatik.uni-goettingen.de
🌍 www.informatik.uni-goettingen.de

First Supervisor: Prof. Dr. Dieter Hogrefe
Second Supervisor: Prof. Dr. Parisa Memarmoshrefi

Chencheng Liang

I hereby declare that I have written this thesis independently without any help from others and without the use of documents or aids other than those stated. I have mentioned all used sources and cited them correctly according to established academic citation rules.

Göttingen, 05. March 2018

Abstract

Due to the unique characteristics of the wireless acoustic communication channel, such as low communication bandwidth, large propagation delay, multi-path propagation, etc., the terrestrial wireless localization schemes cannot be directly applied to mobile Underwater Acoustic Sensor Networks (UASNs) without modification. Besides, the lifetime of the sensor nodes in UASNs is constrained due to battery re-charging in hundreds of meters below the sea surface is difficult and expensive. Therefore, we propose a power-efficient localization scheme for UASNs and name it Energy Efficient Localization Model (EELM). EELM scheme is designed for sparse and partitioning network scenarios. It formulates the interaction between a sensor node and anchor nodes as a Single-Leader-Multi-Follower Stackelberg game, where the selection of the transmission power is the selection of the strategy. In the game, a sensor node acts as a leader, while the anchor nodes act as followers. The payoff for the sensor nodes consists of the proper transmission power to get itself localized and the corresponding cost (energy consumption of that transmission power). The payoff for the anchor nodes consists of the proper transmission power to serve more sensor nodes and the corresponding cost (energy consumption of that transmission power). We prove that all players select best responses and end up in a socially optimal Stackelberg Nash Equilibrium.

In EELM, we need to decide the weights in utility functions of EELM depending on the deployment and other environmental factors, and improper weights may lead to poor performance. Therefore, we propose an extended version of EELM. It replaces the pair of utility functions with a pair of FISs. The FISs are trained by neuro-adaptive learning method using the data collected from EELM. We name it Energy Efficient Localization Model based on adaptive neuro-fuzzy inference systems (EELM-Fy).

The simulation results show that the energy consumption of EELM and EELM-Fy in the sensor nodes is up to 3.5 times lower than that of the state-of-the-art scheme. And the average energy consumption per node of EELM and EELM-Fy is up to 66% lower than that of the state-of-the-art scheme.

In practice, we use NS-3, an open source software providing a discrete-event network simulator, to simulate the underwater communication environment, In NS-3 we build EELM and EELM-Fy communication scheme. We use Matlab fuzzy logic toolbox to train FISs and embed the FISs into NS-3 by FuzzyLite.

Contents

Chapter 1

Introduction

"Sometimes it is the people who no one imagines anything of, that do the things that no one can imagine."

Alan Mathison Turing

1.1 Motivation

UASNs can be adopted in a broad range of applications, such as environmental monitoring, disaster prevention, oil mining, and military use. Location information of sensor node is necessary for certain applications. And for some applications, location information can yield more profits. Localization techniques have been researched intensively in terrestrial wireless sensor networks (WSNs). However, due to the unique characteristics of acoustic communication channel [1, 2], such as low link quality and long latency (detail described in 2.3), Localization schemes for WSNs cannot be directly applied into UASNs without modification.

Many contributions have been done for underwater localization (see survey [3–5]), and they considered various scenarios but the scenario of sparse and partitioned deployed UWSNs. In many UWSN applications, such as underwater surveillance, the nodes are sparsely deployed due to the high deployment cost. Therefore a scheme Opportunistic Localization by Topology Control (OLTC) [6] is proposed for node localization in sparse and partitioned deployed UWSNs. OLTC formulates the interaction between a sensor node and anchor nodes as a Single-Leader-Multi-Follower Stackelberg game. The gaming process is used to control the transmission power of nodes because the energy of sensor nodes are costed mainly by transmitting, which is much larger than that of the idle listening and receiving consumption [7]. The results of the simulation in OLTC shows that in some circumstances, the localization coverage and energy efficiency of OLTC are higher than that of 3DUL (Three-Dimensional Underwater Localization [8]) and DNRL (Dive and

Rise Localization) Protocol [9]. But OLTC has following disadvantages:

1. The gaming process results that sensor node always uses the maximum transmission power to broadcast the “Request" message, which results in more energy consumption of the sensor nodes. However, in many scenarios of UWSNs, energy-saving for sensor nodes is much more important due to the limited battery.

2. Anchor nodes can adjust their transmission power to send “Reply" message for saving energy. However, anchor nodes may be any device (e.g. ship)floating on the water, they are rechargeable or have a large amount of energy. It is better to cost more energy in anchor nodes to serve sensor nodes.

3. Energy consumption is not considered in utility functions in both sensor and anchor nodes. In another word, there is no variable of the energy consumption in utility functions. In utility functions, OLTC does not consider saving energy while it considers how to improve localization coverage.

Therefore, we propose EELM scheme. It also formulates the interaction between a sensor node and anchor nodes as a Single-Leader-Multi-Follower Stackelberg game, but in which both sensor and anchor nodes can adjust their transmission power with the consideration of their own energy consumption by new utility functions. In addition, EELM scheme adds “two-hop" neighbor mechanism (explained in 2.2). This mechanism gives sensor node more information to select a better transmission power, which can raise the probability to localize itself and potentially reduce the energy consumption.

Because in EELM scheme, the weights of utility functions are given manually depending on the deployment and environment. It is subjective, and when we give improper weights the performance of EELM is worse than other schemes. Thereby, we propose EELM-Fy scheme. In general EELM-Fy scheme is the same as EELM scheme, the only difference between them is that EELM-Fy uses a pair of FISs to replace its utility functions. The FISs are generated by neuro-adaptive learning method and the training data is generated from EELM scheme running in the given scenarios with random weights. Therefore, one can run EELM in the given scenarios and collect the data, then train the FISs with the collected data, and get an EELM-Fy scheme eventually. This process is objective and requires no knowledge of the environment. In summary, EELM-Fy is an intelligent version of EELM by combining fuzzy logic control and neuro-adaptive learning. Note that EELM-Fy cannot be used independently because EELM-Fy is trained the data collected from EELM.

In summary, the contributions of this paper are given as follows:

1. By adding a Single-Leader-Multi-Follower Stackelberg game in the scheme, the anchor and the sensor nodes can explore their best strategy to maximize their payoffs. Eventually the Stackelberg Equilibrium is achieved, in which either leader or followers in the game cannot

improve their payoffs by unilateral modification of the strategy.

2. By adding “two-hop" neighbor mechanism, sensor nodes gain more information, which can improve probability to get better strategy in the game. And this fits the purpose of saving more energy for sensor nodes.

3. By using fuzzy logic control and neuro-adaptive learning, one can use EELM to generate EELM-Fy scheme, which does not need subjective decisions and understanding of the environment.

A typical application of EELM is event-driven ocean surveillance with disposable sensor nodes. For instance, there are 10 sensor nodes deployed in the certain underwater area, and sensor nodes only carry the energy, which can support 10 times localization process. And in most of the time, the sensor nodes are in the state of sleep. When we need the location information of the sensor nodes, we instruct anchor nodes to drive EELM process first. Anchor nodes and sensor nodes do not need to keep sending messages to each other.

1.2 Related Work

All thoughts and concepts proposed in EELM are inherited and modified from the schemes we mentioned in this section. We explain the related schemes as follows:

- Dive‘N‘Rise Localization (DNRL) [9] is a distributed localization protocol. The anchor node is called Dive‘N‘Rise (DNR) beacons, and it can descend and ascend in the water by the hydraulic principles. When DNR beacons float on the surface, they can use GPS to localize themselves. After they get their own coordinations on the surface, they descend to send their coordinations to help other nodes localize themselves. DNR beacons periodically descend and ascend until the mission is over. When the sensor nodes receive messages from DNR beacons, they can use one-way ranging ToA technique to calculate the distances to DNR beacons. With the distances and coordinations of DNR beacons, sensor nodes can use lateration to calculate their location.

- Multi-Stage Localization (MSL) [10] is a distributed localization scheme modified from DNRL. In MSL, after the sensor node localizes itself by messages from DNR beacons, it transforms to an anchor node to help other sensor nodes for localization. The main drawback of this scheme is that the localization error accumulates during the iterative localization process.

- Large-Scale Hierarchical Localization (LSHL) Protocol [11] is a hierarchical localization scheme for stationary UASN. It uses 3 types of nodes: “surface buoys", “anchor nodes" and “ordinary sensor nodes". Surface buoys localize themselves by GPS. LSHL assumes that anchor nodes are localized by surface buoys at an earlier deployment stage. Anchor nodes broadcast their coordination to help ordinary nodes localize themselves. If an ordinary node

receives 3 anchor nodes' coordinations and calculates the distances by one-way ranging ToA method, it can perform lateration to estimate its location. Ordinary sensor nodes may transform to anchor nodes when their confidence value is above a certain threshold. LSHL also has two-hop neighbor concept, and it use an extended Euclidean distance estimation method to estimate the distance between the node and its two-hop neighbor. The authors of LSHL mention that LSHL has the highest energy consumption and the highest overhead comparing to DNRL and MSL [12].

- Three Dimensional Localization Algorithm for Underwater Acoustic Sensor Networks (3DUL) [8] is an distributed iterative localization scheme. it needs 3 anchor nodes to initialize the process, and the anchor nodes can localize themselves by GPS signal. Sensor nodes measure the distance from anchor nodes by two-way ranging ToA technique. When sensor nodes find three anchor nodes, it projects the location of the anchors on its plane and estimates self location via lateration. When the sensor node localize itself, it transforms to an anchor node, and the above process continues iteratively similar to MSL and LSHL.

- Underwater Positioning System (UPS) [13, 14] is a TDoA-based localization scheme for stationary UASNs extended from WSN localization scheme [15]. UPS employs four fixed anchors, and the sensor nodes know the locations of these anchor nodes. The anchors send beacon signals to each other sequentially. The sensor nodes covered by four anchor nodes can hear these beacons and calculates the TDoA between the beacons. With the locations of anchor nodes and the TDoA values, the sensor nodes can estimate their locations.

- Localization Scheme for Large Scale underwater networks (LSLS) [16] adds iterative localization phase and complementary phase to UPS. Initially the sensor nodes localize themselves by UPS. In the iterative phase, some localized sensor nodes transform to anchor nodes and repeat UPS scheme to help other unlocalized sensor nodes to localize themselves. In the complementary phase, unlocalized sensor nodes select different set of anchor nodes and repeat UPS scheme to localize themselves. In localization coverage, LSLS performs better than UPS. LSLS can be employed in large-scale UASN with short-range acoustic communications. However, due to iterative and complementary phases, communication overhead and energy consumption of LSLS is higher than that of UPS.

All these schemes consider various constraints of UWSN, but in sparse and partitioned deployment scenarios, where the sensor nodes lack the required number of reference (anchor) nodes, many localization schemes exhibit limited localization coverage. OLTC is proposed to solve this problem, but OLTC is not energy-efficient enough, so we propose EELM scheme, which is more energy-efficient than OLTC while the localization coverages for both are close.

1.3 Thesis Organization

The thesis is organized as follows.

- Chapter 2 explains some basic concepts, such as Stackelberg game, dynamic transmission range in UASNs, challenges of underwater communications, localization techniques for UASNs, and fuzzy inference system, to help understand EELM and EELM-fy.
- Chapter 3 introduces EELM process and how the Stackelberg game works in EELM. We prove the existence of Stackelberg equilibrium.
- Chapter 4 introduces how to transform EELM to EELM-Fy. It described how to prepare train from EELM, how to train FISs, and how to embed FISs into NS-3.
- Chapter 5 describes the simulation settings, assumptions, message formats, and characteristics of the NS-3 UAN model, which are used to implement EELM.
- Chapter 6 presents simulation results of five models (EELM, EELM-Min, EELM-Max, EELM-Fy, OLTC). We also analyze and compare the performances of these models.
- Chapter 7 summarizes the work and provides some concluding remarks and options for further work.

Chapter 2

Basics

"Initial deviate from the truth, and in the end will be one false step will make a great difference."

Aristotle

2.1 Stackelberg Game

Stackelberg game [17] is first proposed in economics. The leader usually is the oligopolistic firm in the market, and the followers are small firms. They make strategies (decide prices in a simple model) in the gaming process to maximize their benefits. In a single-leader and multiple-follower stackelberg game, the leader always move first, then the followers respond to the movement of the leader in order to maximize their benefits. It seems that the followers have more advantages because they move later, and they can make strategies against the leader. However, at the angle of leader, the leader knows the way that followers respond to its movement. Then based on this knowledge, the leader makes a strategy (movement), and it can predict the reactions from followers. In a sense, the leader can manipulate the movements of followers to maximize its own benefit through moving first, and the followers respond to the movement of leader to maximize their benefits. When all players choose optimized strategies to maximize their benefits, a Stackelberg Equilibrium (SE) of the game is constituted.

We use a simple example [18] given in Table 2.1 to illustrate the concept of single-leader and multiple-follower Stackelberg game. Note that this example is just for the illustration of an SE and not an instance of the problem studied. The first number in each cell is the payoff of the leader, while the second is the payoff of the follower. The leader knows following information:

- If the leader plays strategy U, which can get payoff 3 or 6 according to the response of the follower. The follower can choose to play strategy L with payoff 2 or R with payoff 5. Because

the follower wants to maximize its own benefit, the follower would play strategy R. By this choice of the follower, the leader gets payoff 6.

- If the leader plays strategy D, which can get payoff 4 or 8 according to the response of the follower. The follower can choose to play strategy L with payoff 3 or R with payoff 2. Because the follower wants to maximize its own benefit, the follower would play strategy L. By this choice of the follower, the leader get payoff 4.

Hence leader would choose to play strategy U, as it can yield higher payoff (6) comparing with strategy D which yields payoff 4. The premise of this result is that the leader knows when it plays strategy U, the follower will definitely play strategy R. Therefore the Stackelberg Equilibrium of this game is (U, R).

Table 2.1: Payoff matrix for a stackelberg game

		Follower	
		L	R
Leader	U	3,2	6,5
	D	4,3	8,2

In EELM, we model the power control problem sensor and anchor nodes as a Stackelberg game. Sensor nodes are the leaders, and anchor nodes are the followers. The strategy of each player is its transmission power. The utility functions of sensor and anchor nodes are defined in Equation 3.1 and 3.2. The Existence of Stackelberg Equilibrium in EELM is proved in Section 3.6.

2.2 Dynamic Transmission Range in UASNs

Figure 2.1 and 2.2 describe concepts regarding dynamic transmission range, “one-hop" neighbor, and “two-hop" neighbor. These concepts are the basics to understand EELM process in Figure 3.1 and EELM data flow in Figure 3.2. There are one sensor node SN_1 and two anchor nodes AN_1 and AN_2. Suppose the minimum and maximum transmission ranges for anchor nodes and sensor nodes are the same. The transmission range is in proportion to transmission power. Suppose AN_1‘ID is 1, AN_2‘ID is 2.

In Figure 2.1, R_{min} and the solid circle denote the minimum transmission range for SN_1, and R_{max} and the dashed circle denote the maximum transmission range for SN_1. OA denotes the opportunistic area for SN_1. Nodes can change their transmission powers to reach any point in the opportunistic area (the area is between R_{min} and R_{max}). Due to the transmission ranges for anchor nodes are the same as that for the sensor node, AN_1 and SN_1 are in R_{min} for each other. AN_2 and SN_1 are in OA for each other. AN_2 and SN_1 are in R_{min} for each other.

In Figure 2.2, suppose that the distance between SN_1 and AN_1 is a, the distance between AN_1

and AN_2 is b , and the distance between SN_1 and AN_2 is c. When AN_1 and AN_2 use minimum transmission power, which can reach R_{min}, to send a wake-up message (also called “one-hop" message and format described in 5.4), AN_1 and AN_2 can receive messages from each other and be aware the distance and exitance between each other, and they will store the neighbor information in neighbor table, while SN_1 can only receive messages from AN_1 and be aware of the distance and existence of AN_1. This is the first broadcast in EELM.

In summary, AN_1 is “one-hop" neighbor for SN_1, and at this point of time, SN_1 has no aware of the existence of AN_2.

In the second broadcast of EELM, AN_1 and AN_2 broadcast “two-hop" messages (format described in 5.5) also using minimum transmission power. Apart from sender‘s information, “two-hop" message also includes the neighbor information of the sender. For example, “two-hop" message from AN_1 contains neighbor table information [ID (2), b] and information of itself [ID (1), sending time]. When SN_1 receive the “two-hop" information from AN_1. SN_1 is aware of that AN_1 is a m far from me, and AN_2 is out of R_{min} but AN_2 is in R_{min} of AN_1. Notice that SN_1 cannot receive any message from AN_2, because AN_2 uses minimum transmission power to send messages, and the distance between AN_2 and SN_1 is over more than R_{min}. So, at this point of time, SN_1 is aware of that the distance between AN_2 and SN_1 is c and $a < c <= a + b$ (AN_2 may exist on any position of the fan-shaped bold line).

In summary, since SN_1 is aware of AN_2 from AN_1, we call AN_2 as “two-hop" neighbor for SN_1.

We take $c = a + b$ as an estimated distance between AN_2 and SN_1, even though $a < c <= a + b$, because in a triangle, the sum of two edges always larger than the third edge. In a sense, even though this estimation may cost more energy consumption in sensor nodes (further distance needs higher transmission power to reach), it is better than that the sensor nodes cannot localize themselves. For example, Suppose SN_1 needs 2 coordinations from anchor nodes to localize itself. SN_1 known the distance between AN_1 and itself is a from “one-hop" message. SN_1 estimates that the distance between AN_2 and itself is c, and $a < c < a + b$ rather than $c = a + b$, even if the real distance between AN_2 and SN_1 is more than the estimated c. Then SN_1 will send “Request" message (format described in 5.6) with the transmission power which can just reach distance c. Actually, the “Request" message only reach AN_1. AN_2 will never send “Reply" message (format described in 5.7) back to SN_1. Eventually, SN_1 can only receive one or none “Reply" message, and it cannot localize itself.

In summary, we use the estimation $c = a + b$ in order to prevent this kind of situation.

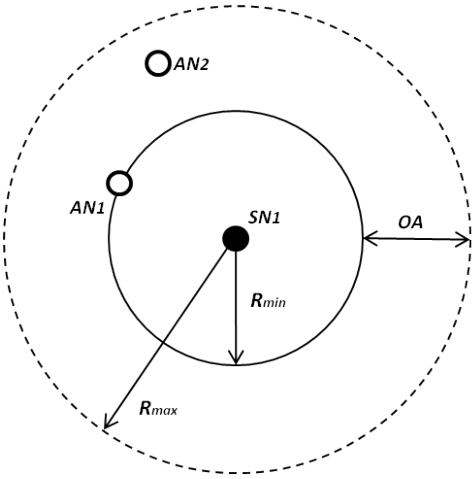


Figure 2.1: Dynamic transmission range for SN_1

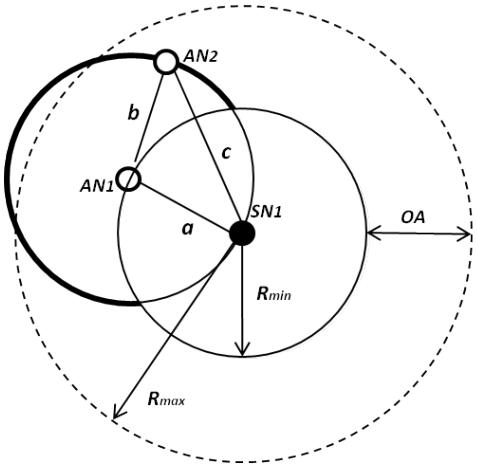


Figure 2.2: “one-hop" and “two-hop" nodes for SN_1

2.3 Challenges of UASNs and Underwater Communications

2.3.1 Propagation Delay

Underwater communication suffers from large propagation delay due to low bandwidth and propagation speed. Low bandwidth also poses loss of network connectivity and network topology changes. The bandwidths corresponding to transmission ranges can be found in Table 2.2 [19, 20].

Table 2.2: The relation between bandwidth and transmission range

Range	Range/km	Bandwidth/kHz
very long	1000	<1
long	10-100	2-5
medium	1-10	10
short	0.1-1	20-50
very short	<0.1	>100

The low bandwidth leads to low data rates. For most acoustic modems data rates is up to 20 Kbps [21–24]. In the short-range underwater communication (under 100 m), data rates can reach Mbit/s in very clean water conditions [25]. However, common ocean environments are very complex, besides, the optical signals suffer from absorption and scattering in the long-range propagation. In practical, usually optical modems are used in 5-10 m communication range [26].

The propagation speed of the acoustic signal is almost five orders of magnitude lower than radio communications. The speed of sound in water increases with increasing pressure, temperature and salinity, which vary with depth and location [27]. The speed of sound in water varies between

1450 and 1500 m/s. This large propagation delay (0.67-0.68 s/km) can reduce the throughput of the system considerably.

2.3.2 Multi-Path Propagation and Time-variability

When a source node launches a beam of rays, each ray will follow a slightly different path, and a receiver placed at some distance will observe multiple signal arrivals. The delay of arrival can range from a few milliseconds to several hundreds of milliseconds. The major reason for multi-path propagation is the reflections from the ocean surface, bottom, and any objects. Obeying Snell's law, a ray of sound always bends toward the region of lower propagation speed. Note that when the ray of sound travels a long path, the signals with undirect paths may arrive first, and the signal with the direct path to the receiver may arrive later.

Multi-path phenomenon generates Inter-Symbol Interference (ISI), which causes severe degradation of the acoustic communication signal [28]. However, many signals echoes had multiple reflections lost much energy can be discarded. There are only some significant paths with strong signal left.

Channel's time variability is caused by inherent changes in the propagation medium and motions of nodes. Some changes of medium, such as temperature, may change in a long timescale, they do not influence the signal significantly. However, some changes of medium, such as surface waves, which effectively cause the displacement of the reflection point, resulting large influences in signals [29].

2.3.3 Doppler Effect

In common communication system, the motion of transmitter or receiver brings the change in frequency or wavelength of wave for the transmitter or receiver. This phenomenon is called Doppler effect or Doppler shift. The magnitude of the Doppler effect can be calculated by

$$a = \frac{c}{v} \tag{2.1}$$

where c is the relative transmitter-receiver velocity and v is the phase velocity of the wave [29]. Compare to the speed of electro-magnetic waves used in terrestrial radio communication system, the speed of sound used in water acoustic communication is very low. Doppler distortion from water acoustic communication is higher than radio communication system.

For comparison, Suppose a sonic aircraft is communicating with the stationary military base, the speed of the sonic aircraft is 300 m/s, we have $a = 10^{-6}$. In this condition, the Doppler spreading can be neglected. However, in UASNs, except intentional movement, motion of underwater equipment is influenced by drifting with waves, currents, and tides, which may reach a few meters

per second. We suppose a node experiences an unintentional motion at 2 m/s, $a = 1.3 \times 10^{-3}$. This means that even if the underwater acoustic communication system is stationary, the Doppler effect cannot be ignored. In multicarrier systems [30], the Doppler distortion is much severe, because every subcarrier may experience doppler shift in different level, which causes nonuniform Doppler distortion across the signal bandwidth. But Doppler effect does not only bring distortions to the communication system, it also brings benefits. A research shows that explicit Doppler shift modeling can help detection of multicarrier signals [31].

2.3.4 Energy Limitations

UASN applications can be roughly divided into 2 main categories as long-term nontime-critical aquatic monitoring and short-term time-critical aquatic exploration [20]. The former do not collect real-time data, and the equipment supposed to work underwater for a long time. Therefore, saving energy is a critical issue. EELM is in this category. The latter collects real-time data, the main focus of this kind of applications is efficient data transmission rather than energy saving. In addition, in mobile underwater networks with high propulsion energy costs, minimizing network communication energy is not always an important concern [25].

In most terrestrial radio transmission systems, the power required for transmitting and receiving are approximately the same depending on the time spent in the transmit or receive states. In acoustic system, the power required for transmitting is much greater than the power required for receiving packets. Typical transmission power values are on the order of tens of watts. The transmission power is also in proportional to distance. In receiver side, receiving power values are varying from 100mW to a few watts. Some underwater equipment have the sleep mode, in this state, power consumption is extremely low (usually under 1 mW) [7].

Energy-efficiency is required in long-term nontime-critical aquatic monitoring UASNs because nodes are expected to work in the ocean for several weeks or months before they are collected and recharged for their next mission [3]. For some special missions, considering the cost of collecting equipment (sensors, robots) from the deep ocean, the equipment may disposable. Therefore, saving energy is directly saving the lifetime of the equipment. Currently, energy harvesting underwater sensor nodes are not used in commercial market because of the size and cost limitations. In the future, under water sensor nodes may transform energy from currents, solar radiation or wind power. However, these energy sources are intermittent, thereby, saving energy during transmitting messages to sensor nodes still essential. And this is the major focus in EELM.

There are many ways to save energy in UASNs, for example, WHOI modem [7] raises the bit rate from low rate at 80 b/s to 5 kb/s, the high-rate mode requires about 60 times less energy per bit (18 dB) although it requires 3 W for detection of high-rate signals as opposed to 80 mW for detection of low-rate signals. The difference between high-rate and low-rate signals is 2 dB. Another important

way to conserve energy is to prevent retransmission. In EELM, we focus on saving energy in sensor nodes by optimizing the transmission power using Stackelberg game theory.

2.4 Localization Techniques for UASNs

Localization techniques for terrestrial wireless sensor networks have been researched a lot, the detail can be found in articles [32, 33]. However, due to the challenges in UASNs we described in Section 2.3, GPS-based localization schemes cannot be directly applied to UASNs. GPS-free localization schemes [34, 35] usually bring high communication overhead. Therefore many novel localization schemes have been proposed.

Articles [3, 5] classify localization schemes into centralized and distributed techniques, in which two subcategories are divided into estimation-based and prediction-based schemes.

1. Centralized Localization Techniques: Location of each sensor node is computed in a command center. Only when the command center sends location information to sensor nodes, sensor nodes know their locations.
2. Distributed Localization Techniques: All sensor nodes can localize themselves without central devices.
3. Estimation-based schemes: Sensor nodes care about current locations and always use the most recent information to localize themselves.
4. Prediction-based schemes: Sensor nodes use distance measurements, previous node locations, and anchor locations to predict their locations at next time point.

According to different node mobility models, the survey [4] classifies underwater localization schemes into 3 main categories:

1. Stationary localization schemes: They assume that nodes are fixed on the ocean floor or on surface buoys. The locations of nodes are not influenced by the environment. The nodes' positions are fixed.
2. Mobile localization schemes: They assume that nodes can move by propelled equipment or freely drift with water currents. The nodes always keep moving intentionally or unintentionally.
3. Hybrid localization schemes: Some nodes keep moving intentionally or unintentionally, and some nodes keep stationary on fixed positions.

These categories can cover almost all well-known schemes. We explain a few schemes.

Area-based Localization Scheme (ALS) [36] is a centralized estimation-based stationary scheme, in which sensor nodes passively listen to the messages from anchor nodes, then sensor nodes send

this information to sink node, and the sink node calculates the location by the information sent from sensor nodes.

Collaborative Localization (CL) [37] is a centralized prediction-based mobile scheme, in which nodes are divided into profilers and followers. Both of them can descend into the water with the same speed, and the profilers descend first. The distances between the profiler and the followers are periodically measured with ToA (Time of Arrival). The location of the profiler gives a prediction of the future locations of the followers.

Localization with Directional Beacons (LDB) [38] is a distributed estimation-based hybrid scheme, in which AUV is anchor node. it floats on the surface and receives its coordinates from the GPS, then it dives to a certain depth and does dead-reckoning for self-localization. AUVs use directional acoustic transceivers to broadcast their coordinates and the angles of their transceiver's beam. Sensor nodes receive this information to localize themselves.

Scalable Localization with Mobility Prediction (SLMP) [39] is a distributed prediction-based hybrid scheme. it uses 3 devices: surface buoys, anchor nodes, and ordinary sensor nodes. Surface buoys receive their coordinates from GPS and send the coordinates to anchor nodes. Anchor nodes can use received coordinates form surface buoys or their previous coordinates and their mobility patterns to estimate their locations. Anchor nodes broadcast its coordinates along with predicted mobility pattern. The ordinary nodes use the received mobility pattern and coordinates to predict their locations, and the pattern is assumed to be valid until an update from an anchor node is received.

2.5 Fuzzy Inference System

Fuzzy Inference System (FIS) formulates the mapping from given inputs to outputs using fuzzy logic. The process of a FIS can be found in Figure 2.3. Fuzzification interface unit converts crisp inputs into fuzzy inputs. Decision-making unit performs operations on rules on fuzzy inputs. Defuzzification interface unit converts the fuzzy quantities processed from decision-making unit into crisp quantities. Rule base contains fuzzy IF-THEN rules. Database defines the membership functions of fuzzy sets.

The fuzzy inference process is described sequentially as follows [40]:

1. Fuzzification : It is the process of transforming a real scalar value into a fuzzy value by different types of membership functions. Fuzzy linguistic variables are used to represent qualities spanning a particular spectrum.

2. Apply fuzzy operator: If there are two or more membership values from fuzzified input variables, we use fuzzy operators to process them and get single output. It could be T-Norm

or S-Norm. Examples of T-Norms are the minimum and the algebraic product, whereas examples of S-Norms are the maximum and the algebraic sum.

3. Apply implication method: We assign proper weight to each rule first, then the implication is implemented for each rule. It determines the consequent of rule by combining the rule strength and the output membership function. For example, min (minimum) truncates the output fuzzy set, and prod (product) scales the output fuzzy set.

4. Aggregate all outputs: Every rule generates an output. Aggregation is the process to combine these outputs into a single fuzzy set. The input of the aggregation process is the outputs from the implication process for each rule. The output of the aggregation process is one fuzzy set for each output variable. The example methods are Max, Probor, and Sum.

5. DeFuzzification : The outputs from aggregation are fuzzy sets. However, the desired outputs from FIS are real numbers. The defuzzification process is to transform these fuzzy sets to real numbers. The example defuzzification methods are centroid, bisector, middle of maximum, largest of maximum, and smallest of maximum.

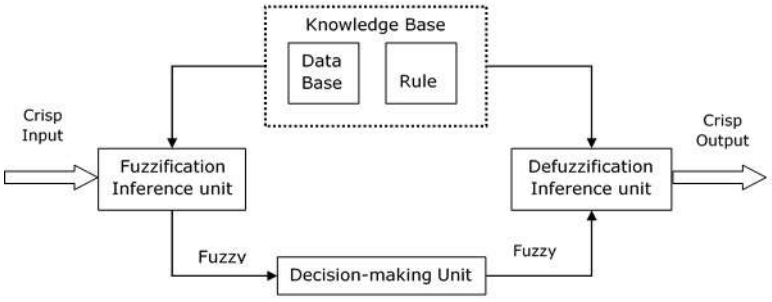


Figure 2.3: Functional Blocks of FIS [41]

2.6 Conclusion

In this Chapter, we introduce some basic concepts, such as Stackelberg game, “two-hop" neighbor, dynamic transmission range, and FIS, to help understand EELM and EELM-Fy scheme explained in following chapters. In addition, we discuss challenges of UASNs and underwater communications, in which the unique characteristics of the acoustic communication channel, such as long propagation delay, multi-path propagation, time-variability, severe doppler effect, and limited energy, are introduced. These characteristics cause the difficulties of localization in UASNs. Besides, we also mentioned some localization techniques for UASNs.

Chapter 3

Energy Efficient Localization Model

"Fantasy is the wings of the poet, hypothesis is the ladder of science."

Johann Wolfgang von Goethe

3.1 Process of EELM scheme

Figure 3.1 shows the overall localization process of EELM. This is only a sketch of EELM because different situations and deployments derive many branches and the corresponding responses in some steps. The details are described in Section 3.2 and 3.3.

In Figure 3.1, there are only two anchor nodes and one sensor node for easier understanding of EELM. In real environment or simulation, there will be more anchor and sensor nodes. In this scene, we suppose all nodes in their minimum transmission range for each other, which means they can receive all messages from each other. The sensor node needs two coordinations from anchor nodes to localize itself. The sequential steps are described as follows:

1. Two anchor nodes generate wake up messages (format described in Table 5.4) including sending time, node information, and flag. They broadcast wake-up messages with minimum transmission power which can reach all nodes in the minimum range around the transmitter (transmission range is described in Section 2.2). We call the first wake up message with Flag value 1 in the head "one-hop" message.

2. The anchor nodes always listen to the acoustic channel. When they receive messages, they check the head of messages. If the flag value is 1, they will store the neighbor information in neighbor table extracted from "one-hop" messages. 2.3. In the sensor node, if it receives a message with flag value 1, it will also store the neighbor information in a neighbor table called T1 extracted from "one-hop" messages. In EELM, sensor and anchor nodes receive different messages. How to store and process these messages is described in 3.2.

3. After anchor nodes receive “one-hop" messages from other anchor nodes or a time slot elapses, anchor nodes generate “two-hop" messages (format described in Table 5.5) including neighbor table stored in step 2, node information, etc. They broadcast “two-hop" messages with minimum transmission power.

4. The sensor node always listens to the acoustic channel. When it receives messages with flag value 2, it will extract information from “two-hop" messages and store the information in a table called T2 (described in 3.2). If the anchor nodes receive messages with flag value 2, they drop the packets and do nothing with these messages.

5. Because there is duplicated and inconsistent information from “one-hop" and “two-hop" message, we need to deduplicate information and merge the information from T1 and T2 into T3 (described in 3.2).

6. Due to T3 has cleaned neighbor information, we use Equation (3.1) to calculate the payoff for each line of data in T3.

7. After the sensor node receives “one-hop" and “two-hop" messages or the time slot elapses, it generates a “Request" message (format described in Table 5.6) including its node message, required number of replies, etc. According to the payoff calculated in step 6, the sensor node selects a transmission power corresponding to the highest payoff to broadcast the “Request" message.

8. If anchor nodes receive messages with flag value 3, they extract information from “Request" message and store the information. If there are other sensor nodes, sensor nodes receive messages with flag value 3, they drop the packet and do nothing with this message.

9. Anchor nodes utilize Equation (3.2) to calculate payoff for each “Request" message received from sensor nodes.

10. After the time slot for receiving “Request" messages elapses, anchor nodes generate “Reply" message (format described in Table 5.7) including its node message, sending time, coordination etc. According to the payoffs calculated in step 9, anchor nodes select optimized transmission powers corresponding to the highest payoff to broadcast “Reply" messages.

11. When the sensor node receives a message with flag value 4, it extract the information from the “Reply" message. When anchor nodes receive “Reply" messages, they drop the packets and do nothing with these messages.

12. With the information (coordinations and sending time) extracted from “Reply" message, the sensor nodes can use certain localization algorithm (trilateration is used in simulation) to localized themselves.

Figure 3.2 shows the data flow in the localization process. There are one sensor node SN_1 and 6 anchor nodes $AN_1, AN_2, \cdots, AN_6$. The solid circle is the minimum transmission range of SN_1,

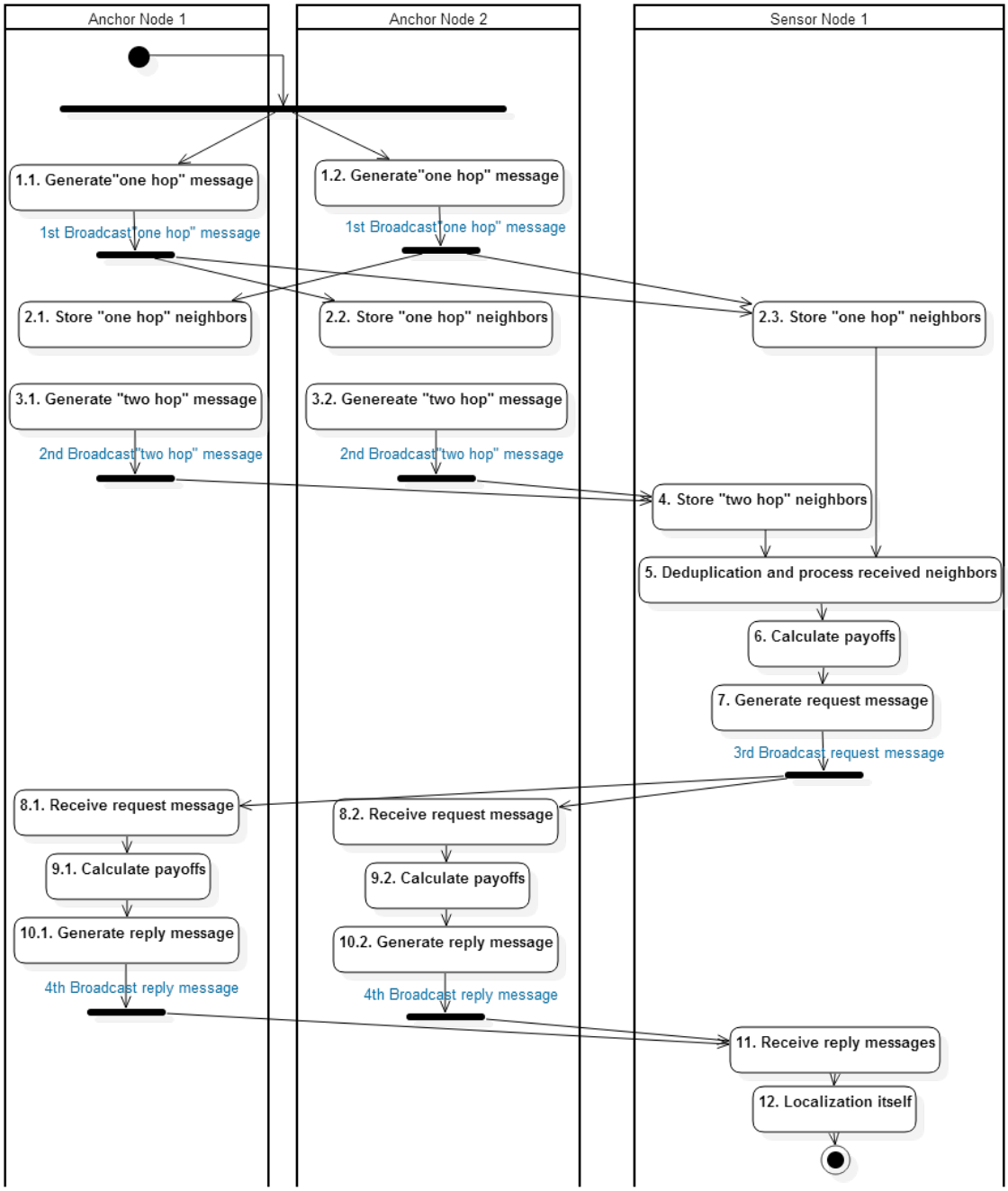


Figure 3.1: Topology control process

and the dashed circle denotes the maximum transmission range of SN_1. Combine Figure 3.1 and 3.2, we describe the data flow in anchor and sensor nodes with 4 broadcasts. The distance in tables is calculated by $S_{sound} \times (T_{receive} - T_{send})$, where S_{sound} is the speed of sound. $T_{receive}$ and T_{send} is the receiving time and the sending time of the message respectively. Suppose the minimum and maximum transmission ranges for anchor nodes and sensor nodes are the same. If there is dashed straight line between anchor nodes, which means the two anchor nodes are in the minimum transmission range for each other. All nodes can differentiate messages from flag value in the message. Thus the nodes can decide to process or drop the message. The ID for $AN_1, AN_2, \cdots, AN_6$ is $1, 2, \cdots, 6$ respectively.

In the first broadcast, all anchor nodes use minimum transmission power to broadcast “one-hop" messages to the acoustic channel (format described in Table 5.4). SN_1 can receives 4 “one-hop" messages from $AN_1, AN_2, \cdots, AN_4$, because these 4 anchor nodes use minimum transmission power which can reach SN_1. SN_1 extracts information from “one-hop" messages and stores ID and distance information in table 1 (T1). The distance can be calculated by sending time of the message. T1 records the “one-hop" neighbor information for SN_1. For instance, we can see that the distance from SN_1 to AN_1 is 1000 m in the first line of T1. In the same time, anchor nodes also receive “one-hop" messages from other anchor nodes. They also extract information from “one-hop" messages and store ID and distance information in neighbor tables (N1-N6) in $AN_1, AN_2, \cdots, AN_6$ respectively. For instance, we can see that the distance from AN_1 to AN_2 is 1000 m in the first line of N1, and the distance from AN_2 to AN_3 is 1300 m in the second line of N2. We only list neighbor tables N1-N4 in Figure 3.2 due to restrained figure space, but N5 and N6 exist. In N5, AN_1 and AN_5 are neighbors, and the distance is 1400 m. In N6, AN_4 and AN_6 are neighbors, and distance is 1200 m.

In the second broadcast, all anchor nodes use minimum transmission power to broadcast “two-hop" messages (format described in Table 5.5) which contain their neighbor tables and their node information. For example, AN_1‘s “two-hop" message contains neighbor table N1, and AN_2‘s “two-hop" message contains neighbor table N2. For anchor nodes, if they receive “two-hop" messages, they drop the packet and do nothing with the message. For SN_1, it receives 4 “two-hop" messages from $AN_1, AN_2, \cdots, AN_4$, because these 4 anchor nodes use minimum transmission power which can reach SN_1. SN_1 extracts information from “two-hop" messages and stores N1-N4 into table 2 (T2).

In the third broadcast, SN_1 deduplicates and merge T1 and T2 into T3. First, copy information in T1 to T3, because data in T1 is reliable. Then check T2, if there is a line of data which exists in T2 and does not exist in T3, combine T2 and N1-N4 to calculate that line of data, then store it into T3. For example, AN_5 is not the direct neighbor of SN_1, and SN_1 does not known the existence of AN_5. However, AN_5 is the direct neighbor of AN_1, and AN_1 is the direct neighbor of SN_1. This relation can be found combining N1 and T2. Thus SN_1 is aware of AN_5, and we can get a rough distance 2400 m (detail described in Section 2.2) between AN_5 and SN_1. After we get $T3$,

SN_1 uses Equation (3.1) to compute payoffs for each line of data in T3. SN_1 selects an optimized transmission power corresponding to the highest payoff. And SN_1 uses this selected transmission power to broadcast a “Request" message (format described in Table 5.6). If there are other sensor nodes which receive “Request" message, they drop this packet and do nothing with this message.

In forth broadcast, If anchor nodes receive “Request" message, they calculate payoffs using (3.2) for each “Request" message. Then each anchor node selects an optimized transmission power corresponding to the highest payoff. They use selected transmission powers to broadcast “Reply" messages (format described in Table 5.7). If anchor nodes receive “Reply" messages, they drop this packet and do nothing with this message. If SN_1 receives “Reply" messages, it uses the information extracted from “Reply" messages to localize itself.

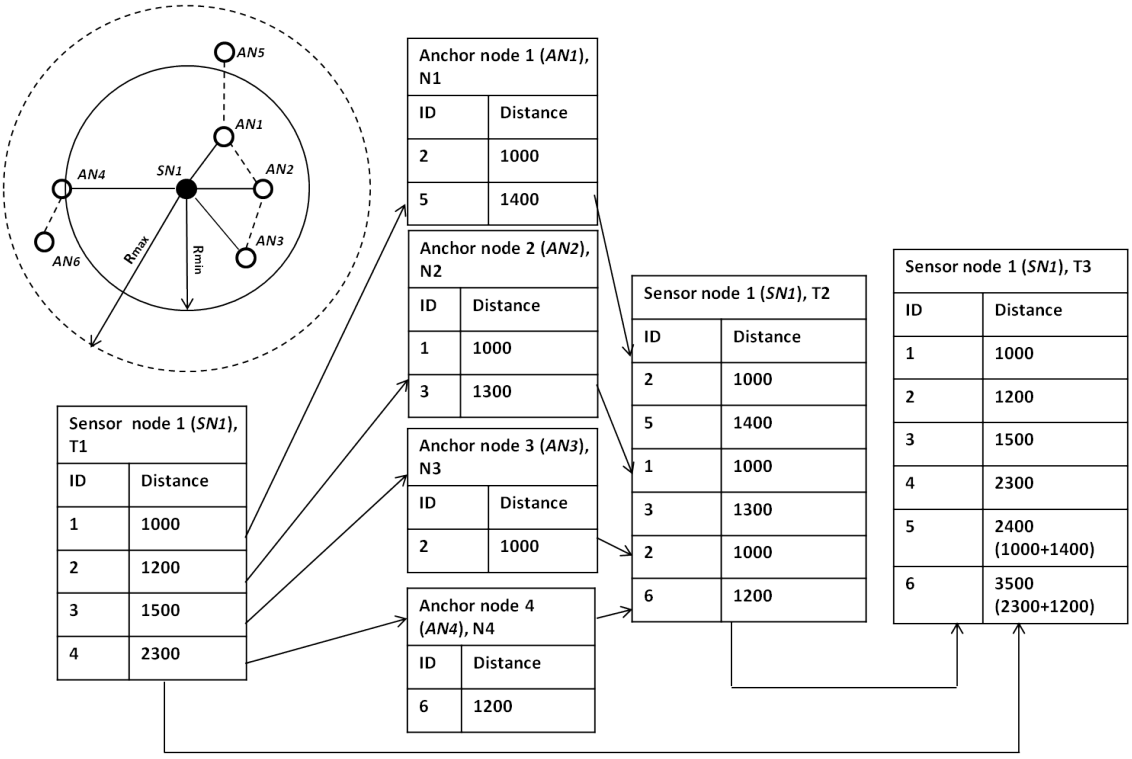


Figure 3.2: Message processing in nodes

3.2 Process of EELM scheme in Sensor Nodes

Figure 3.3 shows the detail of EELM in sensor nodes. Every box with a number means an action.

1. Every sensor node calls function $SetReceiveCallback()$ in class $UanNetDevice$ to be ready to receive packets. If there are messages arrive, jump to action 2. If there is no any message

arrive, jump to action 4, keep listening to the second broadcast (“two-hop" message).

2. If there are messages with the first broadcast (“one-hop") format (Table 5.4) arrive, we can get the anchor node ID and distance between this sensor node and the anchor node. The distance can be calculated by $(T_{current} - T_{message}) \cdot S_{sound}$, where $T_{current}$ is current system time of the sensor node. $T_{message}$ is the time from the message, which is the system time when the anchor node sent this message. S_{sound} is the speed of sound in water, it is discuss in [42], we suppose $S_{sound} = 1500m/s$ here for. We use this formula to compute the distance without consideration of nodal processing delay and transmission delay, because the propagation delay ($1500m/s$) is too slow to consider other delays in millisecond and microsecond. Then we store ID and distance into a table T1, with the format [ID, distance] (explained in Figure 3.2).

3. Determine if there are enough neighbors in T1. In the code, we simply count the number of the rows in T1. We suppose that enough neighbors mean 3 neighbors because we use trilateration for localization. 3.1 if there are enough rows in T1, we use Equation 3.1 to calculate the payoff for every row in T1. Then jump to action 9. If there are not enough rows in T1, jump to action 4.

4. If there are not enough rows in T1, the sensor node keeps listening from the transmission channel. The anchor node, which receives first broadcast (“one-hop") messages from other anchor nodes, will send “two-hop" message. The process in anchor node is described in 3.4.

5. If there are messages with the second broadcast format (Table 5.5) arrive, we can get the anchor node ID and the neighbor information table of this anchor node. we store the received neighbor information table in table T2 with the format [ID, Distance] (explained in Figure 3.2). Then jump to action 6.

6. T1 and T2 have some repeated information, so we deduplicate the repeated rows, and merge T1 and T2 to T3. If T1 and T2 contains the information with the same anchor node ID but different distance respectively, in the deduplication process, we use T1’s information in T3, because T1 contains accurate information. Then jump to action 7.

7. In the code, we count the number of rows in T3 to get the number of neighbor anchor nodes. 7.1 if T3 does not have enough rows (less than 3 rows), the sensor node cannot localize itself due to lack of reference nodes. The process ends. If T3 has enough rows, jump to action 8.

8. If T3 has enough rows, we use Equation 3.1 to calculate the payoff for every row in T3. Then jump to action 9.

9. Sort the calculated payoffs, and choose the transmission power with the highest payoffs. By doing so the sensor node can gain the best profit for itself (ask replies from as many anchor nodes as possible and save as much energy as possible as well). Then jump to action 10.

10. Determine if transmission power can reach enough (3) anchor nodes. We do this, because in some cases, due to the nature of Equation 3.1, the sensor node cannot reach enough anchor nodes by using the transmission power selected by the highest payoff, although there are enough rows in T1 or T3. 10.1 when the chosen transmission power cannot reach enough anchor nodes, transmission power will be raised to reach enough anchor nodes. When the chosen transmission power can reach enough anchor nodes, jump to action 11.

11. If the sensor node can reach enough anchor nodes by the chosen transmission power, the sensor node will broadcast a “Request" message (format is described in Table 5.6) using the chosen transmission power. Then jump to action 12.

12. After broadcasting a “Request" message, the sensor node listens to the channel, and prepare to receive replies from anchor nodes. Then jump to action 13.

13. Determine if the sensor node receives enough “Reply" messages (format is described in Table 5.7) for localizing itself. 13.1 there is not enough “Reply" messages received, so the sensor node cannot localize itself. The process ends. 13.2 the sensor node receives enough “Reply" messages, then it can localize itself by trilateration algorithm. The process ends.

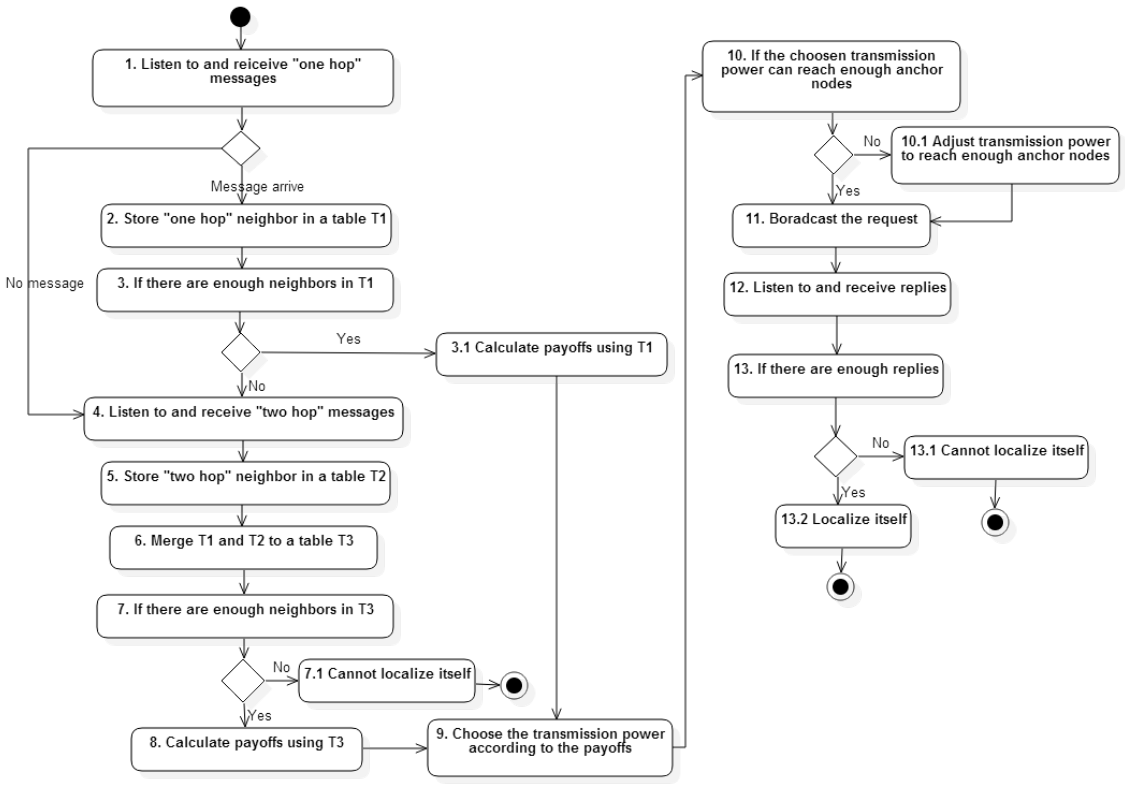


Figure 3.3: Topology control process in sensor nodes

3.3 Process of EELM scheme in Anchor Nodes

Figure 3.4 shows the detail of EELM in anchor nodes. Every box with a number means an action.

1. Every anchor node broadcasts a “one-hop" message by using initial transmission power. Then jump to action.
2. Every anchor node listens to the channel and receives “one-hop" message (described in Table 5.4) sent from other anchor nodes. If there is “one-hop" message arrived, jump to action 3. If there is no message arrive, jump to action 5.
3. Every anchor node stores “one-hop" information to a neighbor table (explained in Figure 3.2).
4. Every anchor node writes the information in neighbor table into “two-hop" message and broadcasts “two-hop" message by using initial transmission power.
5. Every anchor node listens to the channel and receives “Request" messages from sensor nodes. Then jump to action 6.
6. Every anchor node checks the number of received “Request" messages, 6.1 if there is no “Request" message arrived, the anchor node will not send “Reply" message to the channel. If there is only one “Request" message received, jump to the action 9. If there are more than 1 “Request" message arrived, jump to action 7.
7. Calculate payoff (Equation 3.2) for each “Request" message. Then jump to action 8.
8. The anchor node selects an optimal transmission power corresponding to the highest payoff, so it can gain the best profit for itself (respond to as many sensor nodes as possible and save as much energy as possible as well). Then jump to action 9.
9. If there is only one “Request" message received, the anchor node will send a “Reply" message with the transmission power which can exactly reach the sensor node which sends this “Request" message. If there are multiple “Request" messages, “Reply" message will be broadcasted with the selected transmission power.

3.4 Utility Function for Sensor Nodes

Utility function is a mathematical function expressing the way the payoff is calculated. As the leader in a Stackelberg game, the sensor node should consider how to broadcast the localization request message to reach as many anchor nodes as possible with minimum energy consumption. The payoff of any leader i increases with the decrease in energy consumption, which mainly depends on the transmission power. Also, the payoff increases with the increase of transmission

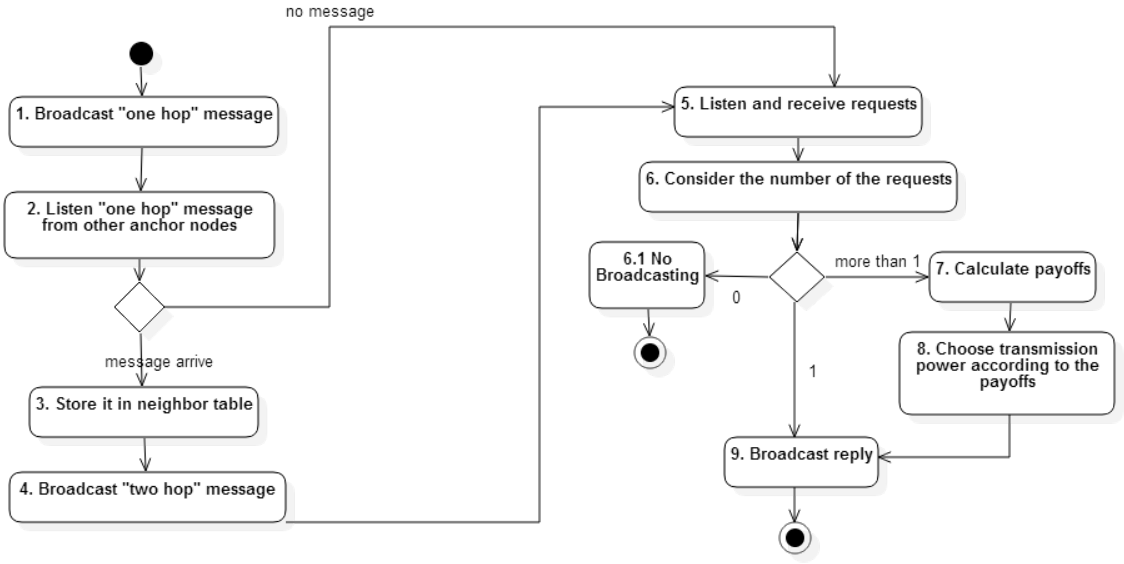


Figure 3.4: Topology control process in anchor nodes

range, because larger transmission range reaches more numbers of anchor nodes, and the potential probability for receiving more “Reply" messages from anchor nodes raises.

Therefore, the utility function for sensor node i is given in following Equation (3.1),

$$U_f(P_i, J_i) = \frac{w_1^i(E_{max}^i - E_i^{P_i} P_i)}{E_{max}^i} + w_2^i(\frac{P_i}{N_i + J_i} + \frac{N_{neig}^{P_i}}{n_{req|max}}) \tag{3.1}$$

where J_i is the transmission power used in the anchor node in order to reach sensor node i, which represents the strategy the anchor nodes may take to against sensor node i. P_i is the transmission power used in the sensor node, which represents the strategy of sensor node i. $N_{neig}^{P_i}$ is the number of neighbor anchor nodes that sensor node i can reach with the transmission power P_i. $E_i^{P_i}$ is the transmission energy cost per unit power of the ith sensor node using transmission power P_i. E_{max}^i is the energy cost per unit power in the ith sensor node using maximum transmission power. $n_{req|max}$ denotes the maximum number of coordinations that the sensor node needs to localize itself. In most localization algorithm, such as trilateration, it is a constant value. N_i is the ambient noise. $w_1^i + w_2^i = 1$, $w_1^i, w_2^i \in (0, 1)$ are the weights of different factors.

3.5 Utility Function for Anchor Nodes

As the follower in a Stackelberg game, the anchor node should consider how to broadcast the localization reply message to reach as many sensor nodes as possible with minimum energy

consumption. The payoff of any follower j increases with the decrease in energy consumption, which mainly depends on the transmission power. Also, the payoff increases with the increase of transmission range, because larger transmission range reaches more numbers of sensor nodes, and the potential probability for serving more sensor nodes raises.

Thus, the utility function of any anchor node j is defined as,

$$U_l(J_j, P_j) = w_1^j \frac{E_{max}^j - C_j^{J_j} J_j}{E_{max}^j} + w_2^j (OA_j - \frac{P_j}{N_j + J_j}) \tag{3.2}$$

where P_j is the transmission power used in the sensor node in order to reach this anchor node j, which represents the strategy the sensor nodes takes to against anchor node j. Jj is the transmission power used in anchor node j, which represents the strategy of anchor node j against the sensor node which uses transmission power P_j to send this localization request information. $C_j^{J_j}$ is the transmission energy cost per unit power of the jth anchor using transmission power J_j. N_j is the ambient noise. E_{max}^i is the energy cost per unit power in the jth anchor node using maximum transmission power. w_1^j and w_2^j are the weights of different factors, $\sum_{k=1}^{2} w_k^j = 1$, $w_k^j \in (0, 1)$. OA_j denotes the localization ability of anchor node j, it is calculated in Equation (3.3).

$$OA_j = \frac{n_{rcv}^{J_j^{'}}}{n_{rcv}} + \frac{n_{rcv}^{J_j^{'}}}{\sum_{k=1}^{n_{rcv}} n_{req}^k} \tag{3.3}$$

where n_{rcv} denotes the number of “Request" messages received from sensor nodes in anchor node j; $n_{rcv}^{J_j^{'}}$ is the number of sensor nodes that the anchor node j can reach with the transmission power $J_j^{'}$. n_{req}^k is the required number of coordination information (from “Reply" messages) for localization in sensor node k. For example, all sensor nodes need 3 coordinations for localizing themselves. Sensor node 1 still needs one coordination to localize itself, and sensor node 2 still needs two coordinations. Thus $n_{req}^1 = 1$ and $n_{req}^2 = 2$. $\sum_{k=1}^{n_{req}} n_{req}^k$ is the overall demanded number of “Reply" messages from sensor nodes.

3.6 Existence of the Stackelberg Nash Equilibrium

When the sensor node (leader) selects the optimal transmission power to localize itself with minimum energy consumption while followers or anchors choose the optimal transmission power to serve the maximum number of sensor nodes with minimum energy cost, the game achieves the equilibrium. In this situation, the payoff of each side cannot be improved by unilaterally changing its own strategy. To find the Stackelberg equilibrium, First, we compute the best response strategies of anchor nodes, for a given strategy of the sensor node. Then we compute the optimal strategy of the sensor node, based on the knowledge of the best response strategies of anchor nodes.

Let $P_i \in [0, P_{max}]$ and $J_j \in [0, J_{max}]$ be the actions available to the sensor node and the anchor nodes, respectively, in which P_{max} and J_{max} mean the actions with maximum transmission power respectively. The sensor node and the anchor nodes make their decision sequentially. The sensor node makes the decision first and then broadcasts the corresponding message to its neighbor anchor nodes. After receiving the message and extracting the strategy from the sensor node, anchor nodes make their individual decision concurrently and non-cooperatively according to their own benefits. The payoffs regarding the sensor node and anchor nodes are interrelated. In other words, anchor nodes' payoffs are contingent on the decision of the sensor node. Let the utility function for the sensor node be $U_l(P_i,\ J_1^{P_i},\ J_2^{P_i}, \cdots,\ J_N^{P_i})$, and the utility function for anchor nodes be $U_{f_1}(P_i,\ J_1^{P_i},\ J_2^{P_i}, \cdots,\ J_N^{P_i})$, $U_{f_2}(P_i,\ J_1^{P_i},\ J_2^{P_i}, \cdots,\ J_N^{P_i})$ $, \cdots,\ U_{f_N}(P_i,\ J_1^{P_i},\ J_2^{P_i}, \cdots,\ J_N^{P_i})$. Then the sensor node gives an action P_i, the Nash equilibrium for the anchor nodes can be defined by the N tuples $(J_{1*}^{P_i},\ J_{2*}^{P_i}, \cdots,\ J_{N*}^{P_i})$.

Formally, the stackelberg equilibrium can be achieved by solving the following optimization problem,

$$\max_{P_i} U_l(P_i,\ J_{1*}^{P_i},\ J_{2*}^{P_i}, \cdots,\ J_{N*}^{P_i}) \tag{3.4}$$

subject to

$$\begin{gathered}
U_{f_1}(P_i,\ J_{1*}^{P_i},\ J_{2*}^{P_i}, \cdots,\ J_{N*}^{P_i}) \geq U_{f_1}(P_i,\ J_1^{P_i},\ J_{2*}^{P_i}, \cdots,\ J_{N*}^{P_i}) \\
U_{f_2}(P_i,\ J_{1*}^{P_i},\ J_{2*}^{P_i}, \cdots,\ J_{N*}^{P_i}) \geq U_{f_2}(P_i,\ J_{1*}^{P_i},\ J_2^{P_i}, \cdots,\ J_{N*}^{P_i}) \\
\cdots \\
U_{f_N}(P_i,\ J_{1*}^{P_i},\ J_{2*}^{P_i}, \cdots,\ J_{N*}^{P_i}) \geq U_{f_N}(P_i,\ J_{1*}^{P_i},\ J_{2*}^{P_i}, \cdots,\ J_N^{P_i})
\end{gathered}$$

where j is integer and $j \in [1, N]$. The sensor node's problem is to determine the optimal decision P_i^* by the objective function (3.4) in the above optimization problem. And the $N+1$ action tuple $(P_i^*, J_{1*}^{P_i},\ J_{2*}^{P_i}, \cdots,\ J_{N*}^{P_i})$ is then the Stackelberg equilibrium of the Single-Leader-Multi-Follower Stackelberg game.

The optimal transmission power p_i^* of sensor node is selected through maximizing its payoff by the objective function (3.4) in the above optimization problem. In order to obtain an equilibrium, the subgame of anchor nodes should have a unique Nash equilibrium for every action announced by the sensor node as well.

In following subsections, we prove the existence of Stackelberg equilibrium in EELM. All symbols without explanations come from Equation 3.1, 3.2, and 3.3. And they are already explained in Section 3.4 and 3.5.

3.6.1 Best Response Strategies of Anchor Nodes

Corresponding to the strategy of the jth anchor node J_j, the transmission power selection problem can be given as an optimization problem shown in Equation (3.7). All anchor nodes are non-cooperative. In Theorem 1, the existence of the best response strategy of each anchor node is proved and the unique equilibrium point is computed.

Theorem 1. *Let J_j be the strategy of the jth anchor node. The detailed computation of the best response of each anchor node is given in Equation* (3.5) *with the conditions given in Equation* (3.6)*,*

$$J_j^*(P_j) = (\frac{E_{max}^j w_2^j P_j}{w_1^j C_j^{J_j}})^{\frac{1}{2}} - N_j \tag{3.5}$$

$$\frac{P_j E_{max}^j}{P_j E_{max}^j + C_j^{J_j} N_j (N_j + J_{max} + J_j) + C_j^{J_j} J_j J_{max}} < w_1^j < 1 + \frac{C_j^{J_j}(J_j - N_j^2)}{E_{max}^j P_j} \tag{3.6}$$

Proof. The optimization problem (3.7) is a standard form of convex optimization problem.

$$\max_{J_j} U_l(J_j) = w_1^j \frac{E_{max}^j - C_j^{J_j} J_j}{E_{max}^j} + w_2^j (OA_{s_j}^j - \frac{P_j}{N_j + J_j}) \tag{3.7}$$

subject to

$$w_1^j = 1 - w_2^j; w_1^j, w_2^j \in (0,1); J_j \in [0, J_{max}]$$
$$P_j \in [0, P_{max}]; C_j^{J_j}, N_j, OA_{s_j}^j, E_{max}^j > 0$$

Because $U_l(J_j)$ is the continuous function in the range $[0, J_{max}]$, the maximum value can be achieved at certain $J_j \in [0, J_{max}]$. The first order partial derivative of $U_l(J_j)$ with respect to the jth anchor node, for $j \in [1, N]$, is

$$\frac{\partial U_l(J_j)}{\partial J_j} = \frac{-w_1^j C_j^{J_j}}{E_{max}^j} + \frac{w_2^j P_j}{(N_j + J_j)^2} \tag{3.8}$$

The second order partial derivative of $f_l(J_j)$ is given in Equation (3.9).

$$\frac{\partial^2 U_l(J_j)}{\partial (J_j)^2} = \frac{-2 w_2^j P_j}{(N_j + J_j)^3} \tag{3.9}$$

Since the value of the second order partial derivative of $f_l(J_j)$ is less than zero, $J_j^{'} = (\frac{E_{max}^j w_2^j P_j}{w_1^j C_j^{J_j}})^{\frac{1}{2}} - N_j$ is the local extremum computed by $\frac{\partial U_l(J_j)}{\partial J_j} = 0$. The maximum value $J_j^*(P_j)$ can be achieved when $U_j \in \{0, J_j^{'}, J_{max}\}$. Let $U_l(0) < U_l(J_j^{'})$ and $U_l(J_{max}) < U_l(J_j^{'})$ with the conditions which are

given in the following equations.

$$
\begin{aligned}
&U_l(0) - U_l(J_j^{'}) < 0 \\
&\Rightarrow w_1^j + w_2^j OA_{s_j}^j + w_2^j \frac{P_j}{N_j} - w_1^j \frac{E_{max}^j - C_j^{J_j} J_j^{'}}{E_{max}^j} - w_2^j OA_{s_j}^j - w_2^j \frac{P_j}{N_j + J_j^{'}} < 0 \\
&\Rightarrow w_1^j < 1 + \frac{C_j^{J_j}(J_j - N_j^2)}{E_{max}^j P(J_j)}
\end{aligned}
$$

$$
\begin{aligned}
&U_l(J_j) - U_l(J_j^{'}) < 0 \\
&\Rightarrow w_1^j \frac{E_{max}^j - C_j^{J_j} J_{max}}{E_{max}^j} + w_2^j OA_{s_j}^j + w_2^j \frac{P_j}{N_j + J_{max}} - w_1^j \frac{E_{max}^j - C_j^{J_j} J_j^{'}}{E_{max}^j} - w_2^j OA_{s_j}^j - w_2^j \frac{P_j}{N_j + J_j^{'}} < 0 \\
&\Rightarrow w_1^j > \frac{P(J_j) E_{max}^j}{P_j E_{max}^j + C_j^{J_j} N_j (N_j + J_{max} + J_j) + C_j^{J_j} J_j J_{max}}
\end{aligned}
$$

□

3.6.2 Best Response Strategy of Sensor Nodes

The existence and uniqueness of Nash equilibrium for the sensor node (leader) is proved in Theorem 2 rigorously. In this section, we use E_i to denote $E_i^{P_i}$, and C_i denotes $C_i^{J_i}$ for simplicity in Equations.

Theorem 2. *Let P_i be the strategy of the ith sensor node. The detailed computation of the best response of each anchor node is given in equation* (3.10) *with the conditions given in Equation* (3.11)*,*

$$
P_i^*(J_i) = \frac{E_{max}^i w_2^i C_i}{4 w_1^i E_i^2} \tag{3.10}
$$

$$
\frac{C_i E_{max}^i (P_{max}^{\frac{1}{2}} - P_i^{\frac{1}{2}})^2}{C_i E_{max}^i (P_{max}^{\frac{1}{2}} - P_i^{\frac{1}{2}})^2 + E_i^2 (P_{max} - P_i)^2} < w_1^i < 1 - \frac{E_i n_{req|max}}{E_{max}^i C_i} \tag{3.11}
$$

Proof. The optimization problem (3.12) is a standard form of convex optimization problem.

$$
\max_{P_i} U_f(P_i) = \frac{w_1^i (E_{max}^i - E_i P_i)}{E_{max}^i} + \frac{w_2^i P_i}{N_i + J_i} + \frac{w_2^i N_{neig}^{P_i}}{n_{req|max}} \tag{3.12}
$$

subject to

$$w_1^i = 1 - w_2^i; w_1^i, w_2^i \in (0,1); n_{req|max} = 3;$$
$$E_i, N_i, E_{max}^i > 0; N_{neig} \geq 0; P_i \in [0, P_{max}]$$
$$J_i = (\frac{E_{max}^j w_2^j P_j}{w_1^j C_j})^{\frac{1}{2}} - N_j \in [0, J_{max}]$$

Because $U_f(P_i)$ is the continuous function on the range $[0, P_{max}]$, the maximum value can be achieved at some certain $P_i \in [0, P_{max}]$. The first order partial derivative of $U_f(P_i)$ with respect to the ith sensor node, is

$$\frac{\partial U_f(P_i)}{\partial P_i} = \frac{1}{2} P_i^{-\frac{1}{2}} (\frac{w_1^i w_2^i C_i}{E_{max}^i})^{\frac{1}{2}} - \frac{w_1^i E_i}{E_{max}^i} \tag{3.13}$$

The second order partial derivative of $U_f(P_i)$ is given in equation (3.14).

$$\frac{\partial^2 U_f(P_i)}{\partial (P_i)^2} = -\frac{1}{4} P_i^{-\frac{3}{2}} (\frac{w_1^i w_2^i C_i}{E_{max}^i})^{\frac{1}{2}} \tag{3.14}$$

Since the value of the second order partial derivative of $U_f(P_i)$ is less than zero, $P_i^{'} = \frac{E_{max}^i w_2^i C_i}{4 w_1^i E_i^2}$ is the local extremum computed by $\frac{\partial U_f(P_i)}{\partial P_i} = 0$. The maximum value $P_i^*(J_i)$ can be achieved when $P_i \in \{0, P_i^{'}, P_{max}\}$. Let $U_f(P(0)) < U_f(P_j^{'})$ and $U_f(P_{max}) < U_f(P_i^{'})$ with the conditions which are given in the following equations.

$$U_f(0) - U_f(P_i^{'}) < 0$$
$$\Rightarrow w_1^i + \frac{w_2^i N_{neig}}{n_{req|max}} - \frac{w_1^i (E_{max}^i - E_i P_i^{'})}{E_{max}^i} - \frac{w_2^i P_i^{'}}{N_i + J_i} - \frac{w_2^i N_{neig}^{P_i}}{n_{req|max}}$$
$$\Rightarrow w_1^i < 1 - \frac{n_{req|max} E_i}{E_{max}^i C_i}$$

$$U_f(P_i) - U_f(P_i^{'}) < 0$$
$$\Rightarrow (\frac{w_1^i (E_{max}^i - E_i P_i^{'})}{E_{max}^i} + \frac{w_2^i P_i^{'}}{N_i + J_i} + \frac{w_2^i N_{neig}^{P_i}}{n_{req|max}}) - (\frac{w_1^i (E_{max}^i - E_i P_{max})}{E_{max}^i} + \frac{w_2^i P_{max}}{N_i + J_i} + \frac{w_2^i N_{neig}^{P_i}}{n_{req|max}}) < 0$$
$$\Rightarrow w_1^i > \frac{C_i E_{max}^i (P_{max}^{\frac{1}{2}} - P_i^{\frac{1}{2}})^2}{C_i E_{max}^i (P_{max}^{\frac{1}{2}} - P_i^{\frac{1}{2}})^2 + E_i^2 (P_{max} - P_i)^2}$$

□

3.7 Gaming Process

In Figure 3.5 and 3.6, we an example to discuss payoffs and the decisions for sensor and anchor nodes in gaming process. $SN_i, (i = 1, 2, 3)$ denotes sensor nodes, $AN_i, (i = 1, 2 \cdots, 6)$ denotes anchor nodes. Suppose minimum and maximum transmission ranges for all anchor nodes and sensor nodes are the same. R_{min} and R_{max} is the minimum and maximum transmission radius respectively. sensor nodes need 3 “Reply" messages to localize itself. Table T3 and N6 are explained in Figure 3.2.

By using different weights in Equation (3.1) and (3.2), the table SP1 and AP6 in SN_1 and AN_6 may have different results. In table SP1 and AP6, we suppose that the payoff values are the values in SP1 and AP6. In the real scenario, calculating payoff is a part of the gaming process according to different information from “one-hop”, “two-hop", and “Request" messages. In another word, data in SP1 and AP6 is hypothetical data we made for easier to explain the gaming process.

In Figure 3.5, we consider at the angle of SN_1, the solid and dashed circle is the minimum and maximum transmission range of SN_1 respectively. SN_1 has the distance information of $AN_1, AN_2, \cdots, AN_6$ extracted from “one-hop" and “two-hop" messages, and all the anchor nodes are in its maximum transmission range. In OLTC model, SN_1 uses the maximum transmission power to broadcast “Request" message, so the message can reach all the anchor nodes. However, it is not necessary to use maximum transmission power to reach all anchor nodes in this case, because if SN_1 only need 3 “Reply" messages to localize itself, it is wasting of energy to use maximum transmission power to broadcast messages and receive more than 3 “Reply" messages. Therefore in EELM, we consider the energy consumption additionally, so we use Equation (3.1), which considers both the energy consumption and the number of required replies, to calculate the payoff. Then SN_1 selects the “Request" message‘s transmission power corresponding to the highest payoff. This transmission power is optimized because it consumes the least energy to reach just enough anchor nodes which may send “Reply" messages back. In table SP1 we see that the highest payoff value is 20, the corresponding transmission power can make SN_1‘ “Request" message reach AN_1, AN_2, AN_3, and AN_4. One may curious that if EELM model is designed to save energy, why the highest payoff does not land on AN_3. In this situation, SN_1 can reach just 3 anchor nodes (AN_1, AN_2, and AN_3), and 3 is the exactly the number of “Reply" message to make SN_1 localize itself. Because the anchor nodes also consider their own energy consumptions in anchor node‘s utility function (Equation (3.2)). the anchor nodes may not respond to all “Request" messages, and we explain it at the angle of AN_6 in Figure 3.6 later.

In the real scenario, the highest payoff value may land on AN_1. In this situation, SN_1‘ “Request" message can only reach one anchor node using the selected transmission power. But according to the gaming mechanism, all anchor nodes (AN_1 to AN_6) may broadcast “Reply" messages with the transmission power which can reach or cannot SN_1. Therefore, SN_1 may localize itself with just 3

or more than 3 “Reply" messages, or SN_1 may not localize itself with less than 3 “Reply" messages.

In Figure 3.6, we consider at the angle of AN_6. The solid and dashed circle is the maximum and maximum transmission range of AN_6 respectively. We can see that in table AP6, AN_6 receives 3 “Request" messages. But according to the payoff values, AN_6 selects the transmission power, which can only reach SN_1 and SN_2, to send “Reply" message. This means that AN_6 does not always serve all sensor nodes due to the consideration of its own energy consumption. Except for the energy consideration, AN_6 chooses not to serve all sensor nodes because other anchor nodes may serve the sensor nodes that AN_6 choose not to serve.

In a brief summary of the gaming mechanism, there are several possible situations. The highest payoffs may land on any line in table SP1 and AP6 according to different weights in the Equation (3.1) and (3.2), and the deployment of nodes. We list all situations may happen in the gaming process:

1. A sensor node sends “Request" message with the transmission power which cannot reach enough number of anchor nodes, the sensor node still receives enough number of “Reply" messages to localize itself.

2. A sensor node sends “Request" message with the transmission power which just can reach enough number of anchor nodes, the sensor node receives enough number of “Reply" messages to localize itself.

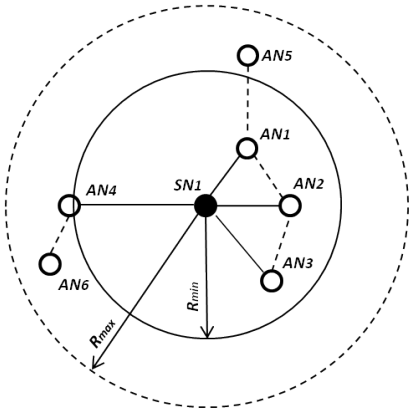
3. A sensor node sends “Request" message with the transmission power which can reach more than enough number of anchor nodes, the sensor node receives enough number of“Reply" messages to localize itself.

4. A sensor node sends “Request" message with the transmission power which cannot reach enough number of anchor nodes, the sensor node does not receive enough number of “Reply" messages, and it cannot localize itself.

5. A sensor node sends “Request" message with the transmission power which can reach just enough number of anchor nodes, the sensor node does not receive enough number of “Reply" messages, and it cannot localize itself.

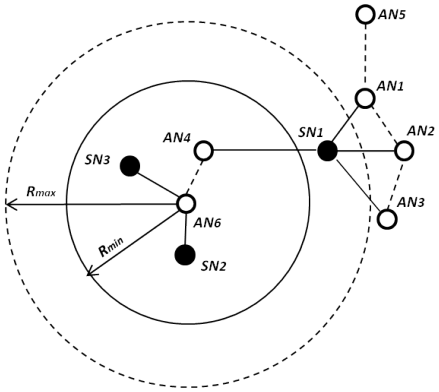
6. A sensor node sends “Request" message with the transmission power which can reach more than enough number of anchor nodes, the anchor node does not receive enough number of “Reply" messages, and it cannot localize itself.

The gaming process may lead to the best situation, in which every node consumes the least energy, and all sensor nodes can localize themselves. It may also lead to the worst situations described in situation 4, 5, 6, in which either the sensor nodes are too selfish to send any “Request" message, or the anchor nodes are too selfish to send any “Reply" message. In the simulation, in order to prevent situation 4, we add a mechanism to ensure enough coverage. The mechanism is described



Sensor node 1 (*SN1*), SP1		
Anchor nodes	Distance (from table T3)	Payoffs
1	1000	12
2	1200	14
3	1500	16
4	2300	20
5	2400	18
6	3500	17

Figure 3.5: Payoffs for SN_1 in gaming



Anchor node 6 (*AN6*), AP6		
Sensor nodes	Distance (from table N6)	Payoffs
1	2400	16
2	1000	20
3	1500	21

Figure 3.6: Payoffs for AN_6 in gaming

before in Figure 3.2, action 10-11. This is only a way to improve simulation result in EELM.

One of the reasons, which leads the worse situations, is that both the utility functions of sensor nodes and anchor nodes (Equation (3.1) and (3.2)) only consider the information in the row-wise of the neighbor tables. They do not consider the global information (column-wise). For example, in sensor nodes, the utility function does not consider the number of the messages that they received. The anchor node' utility function also has this problem. Using wrong weights in utility functions (Equation (3.1) and (3.2)) also leads to the worst situations. In one word, the static mathematical formulas cannot cover all situations To solve this problem, we use a machine learning method to generate two fuzzy inference systems to replace the utility functions. This is introduced in Section 4.

3.8 Conclusion

In this chapter, we introduce the overview of EELM scheme. Then the acting details of sensor and anchor node in EELM scheme are illustrated. We propose the new utility functions of sensor and anchor nodes, which contain the variables of energy consumption. Then with the pair of proposed utility functions, we prove the existence of Stackelberg Equilibrium, in which all nodes select the best strategy and cannot improve their payoffs by unilateral modification of strategy. In the last section of this chapter, we discuss the situations, which may appear in gaming process. And we know that only with the proper weights in EELM, the good situations happen. To solve the weight problem we propose EELM-Fy scheme, which is introduced in next chapter.

Chapter 4

Energy Efficient Localization Model based on ANFISs

"The pursuit of objective truth and knowledge is the highest and eternal goal."

Albert Einstein

The utility functions for sensor and anchor nodes are fixed mathematical formulas, they cannot deal with all special situations properly. And we need to assign weights to the utility functions subjectively according to the scenario. If we assign some improper weights, the performances of EELM are poor. Therefore, we use two FISs to replace utility functions (Equation 3.1 and 3.1). However, as we described in Section 2.5, FIS needs to define all parameters in membership functions, logical operations, and If-Then rules by the user's interpretation of the characteristics of the variables in the model. In another word, we need to decide all parameters manually and subjectively. To set these parameters objectively and automatically, we use Neuro-Adaptive Learning method to generate all parameters of FIS by collections of input and output data. And the training data come from EELM. Because adaptive neuro-fuzzy inference systems are introduced in EELM, we name this model EELM-Fy.

4.1 Adaptive Neuro-Fuzzy Modeling

An adaptive neuro-fuzzy inference system (ANFIS) is a Sugeno-type fuzzy inference system [43] trained by neuro-adaptive learning techniques [44]. Article [45] uses fuzzy logic and neuro-fuzzy method to design two air conditioning systems, and the simulation results show that the Neuro-fuzzy algorithm is definitely superior to the fuzzy logic algorithm as it inherits adaptability and learning.

We use Neuro-Fuzzy Designer, which is an application from Matlab fuzzy logic Toolbox, to design,

train, and test the Sugeno-type fuzzy inference systems. In NS-3, we call Fuzzylite' library [46] to use the trained FISs. Fuzzylite is a free and open-source fuzzy logic control library programmed in C++ for multiple platforms (e.g., Windows, Linux, Mac, iOS).

4.1.1 Training Data set Preparation

We generate training data set from EELM. We run EELM in 2 simulation scenes (Table 5.2), and every time we change the number of anchor or sensor nodes, we run EELM 1000 times. Then we record all inputs and outputs from Equation 3.1 and 3.2 in EELM process. Eventually, we get 78452 lines of data from sensor nodes, which are used to train the FIS of the sensor node in order to replace Equation 3.1. And we get 302658 lines of data from anchor nodes, which are used to train the FIS of anchor node in order to replace Equation 3.2. we use .csv format to store the train data.

Note that we only collect the data, in which the final performances of localization coverage and energy efficiency are higher than the average performances. For instance, we run EELM 1000 times, the average localization coverage is 90%, we record all data. If there are certain times that the average localization coverages are higher than 90%, then we collect these data for training.

In EELM process, when a sensor node uses Equation 3.1 to calculate the payoff. we record the input and output values from Equation 3.1. The ranges of input and output of Equation 3.1 are shown in Table 4.1. Table 4.2 shows some example training data sets recorded from sensor nodes in EELM.

Table 4.1: Inputs and outputs for FIS of sensor node

Input	Range of inputs	Output	Range of output
$N_{neig}^{P_i}$, -	3 - 9		
P_i, dB	111.0.27 - 171.540	payoff, -	0.822-1.882
E_{max}^{i}, dB	138.044 - 156.943		

Table 4.2: Example training data set for FIS of sensor node

$N_{neig}^{P_i}$	P_i	E_{max}^{i}	payoff
3	130.795	144.104	0.847274
4	135.805	151.172	0.863456
5	133.501	156.943	0.882007
6	139.789	156.943	1.37645
7	137.521	156.943	1.37845

In EELM process, when an anchor node uses Equation 3.2 to calculate the payoff. we record the input and output values from Equation 3.2. The ranges of input and output of Equation 3.2 are shown in Table 4.3. Table 4.4 shows some example training data sets recorded from anchor nodes in EELM.

Table 4.3: Inputs and outputs for FIS of anchor node

Input	Range of inputs	Output	Range of output
$N_{rcv}^{P_i}$, -	1 - 26	payoff, -	0.218-8.735
P_j, dB	109.721 - 110.704		
J_j, dB	108.847 - 188.117		
E_{max}^{j}, dB	113.307 - 188.117		
$n_{rcv}^{J_j^{'}}$, -	1 - 26		
$\sum_{k=1}^{n_{rcv}} n_{req}^{k}$, -	1 - 51		

Table 4.4: Example training data set for FIS of anchor node

$N_{rcv}^{P_i}$	P_j	J_j	E_{max}^{j}	$n_{rcv}^{J_j^{'}}$	$\sum_{k=1}^{n_{rcv}} n_{req}^{k}$	payoff
8	110.66	130.655	154.551	1	21	0.343758
6	109.721	149.838	149.838	6	1	3.74185
17	109.877	154.86	169.354	15	24	0.279737
13	110.375	153.228	164.522	9	15	0.274032
15	109.721	134.95	149.917	5	5	0.808755

4.1.2 Training FISs in ANFIS Editor

The training interface (ANFIS editor) in Matlab fuzzy toolbox is shown in Figure 4.1 and 4.2, in which we can see the parameters, such as Epochs and Error tolerance, used to train FIS. The steps of using ANFIS editor to generate FISs are described as follows:

1. Load train data from .csv files into ANFIS Editor.

2. Use subtractive clustering method to generate the FIS. The parameters for subtractive clustering are listed in Table 4.5, they are default setting in Matlab. The structures of generated FISs are shown in Figure 4.3 and 4.4, in which we can see that there are 5 membership functions for every input variable of the sensor node. There are 5 rules generated automatically for the FIS of the sensor node. there are 3 membership functions for every input variable of anchor node. There are 3 rules generated automatically for the FIS of the sensor node. All logic operations in rules are AND.

3. After the FISs are generated, press “Train Now" to train the FISs then we can see the training error in every epochs.

4. Now we can export the trained FISs to .fis files. These files will be used in Fuzzylite, and the Fuzzylite can generate corresponding C++ code which can be embedded into NS-3 code.

Eventually we get two .fis file: anchor.fis and sensor.fis. We name the FIS used in sensor nodes FIS-S and the FIS used in anchor nodes FIS-A. FIS-S and FIS-A is used to replace Equation 3.1 and 3.2 respectively.

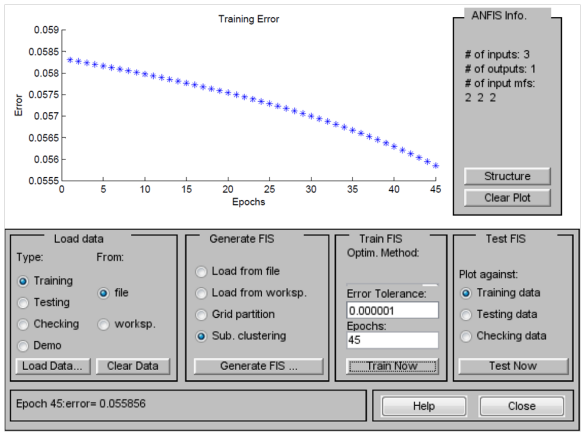


Figure 4.1: Training error of ANFIS for sensor nodes

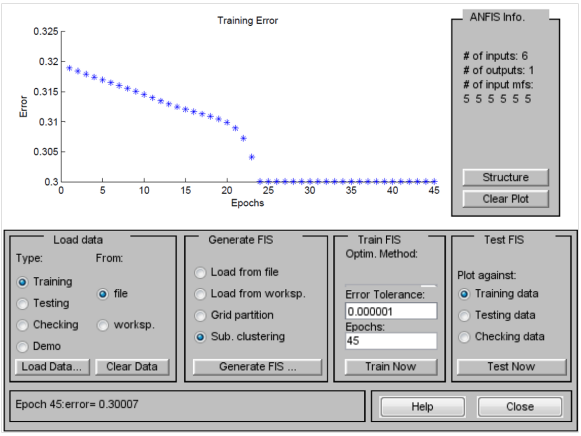


Figure 4.2: Training error of ANFIS for anchor nodes

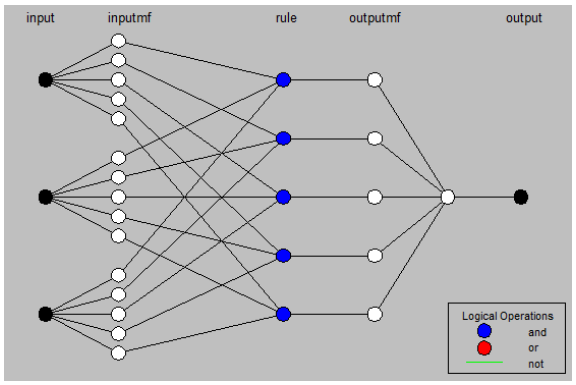


Figure 4.3: ANFIS structure of sensor node

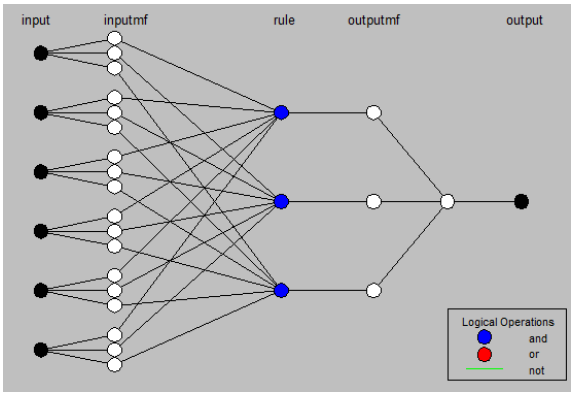


Figure 4.4: ANFIS structure of anchor node

Table 4.5: parameters for subtractive clustering

Range of influence	Squash factor	Accept ratio	Reject ratio
0.5	1.25	0.5	0.15

4.1.3 Embedding FISs Into NS-3 Code by Fuzzylite

We use QtFuzzylite, which is a visualizing tool for Fuzzylite, to open the .fis files generated by Matlab. The input variables for FIS-S are shown in Figure 4.6, in which in1 - in3 is $N_{neig}^{P_i}$, P_i, and E_{max}^{i} respectively. The input variables for FIS-A are shown in Figure 4.5, in which in1 - in6 is $N_{rcv}^{P_i}$, P_j, J_j, E_{max}^{j}, $n_{rcv}^{J_j^{'}}$, and $\sum_{k=1}^{n_{rcv}} n_{req}^{k}$ respectively. The linguistic terms are generated automatically by clustering method, so the names of linguistic terms are cluster 1 - cluster 5 for FIS-S, and the names of linguistic terms are cluster 1 - cluster 2 for FIS-A. And all the membership functions of linguistic terms are Gaussian.

The output variables for FIS-S and FIS-A are shown in Figure 4.7 and 4.8, in which out1 is the payoff for sensor node or anchor node. For both of them, the aggregation method is Maximum, and the defuzzification method is WeightedAverage.

The rule blocks for FIS-S and FIS-A are shown in 4.9 and 4.10. FIS-S has 5 rules, while FIS-A has 2 rules. Because these rules are generated automatically by ANFIS Editor from training data set, all logic operations are AND, and rules are expressed by the generated names (in1 -in6, out1, cluster1 - cluster 5). For both rule blocks for FIS-S and FIS-A, conjunction method is AlgebraicProduct, disjunction method is AlgebraicSum, implication method is AlgebraicProduct, and activation is general.

The steps to replace Equation 3.1 and 3.1 with FIS-S and FIS-A are shown as follows:

1. QtFuzzylite can transform FISs to C++ code. We use QtFuzzylite open sensor.fis and anchor.fis, then export sensor.fis and anchor.fis as C++ code. We encapsulate the transformed C++ code into two functions for convenience of programming and name them FIS-S-C++ and FIS-A-C++ for sensor node and anchor node respectively.

2. In order to call FIS-S-C++ and FIS-A-C++ generated from Qtfuzzylite in NS-3, We need to link fuzzylite library to NS-3 by editing the configuration in the wscript file.

3. In NS-3 code, we can use the encapsulated functions FIS-S-C++ and FIS-A-C++ to replace Equation 3.1 and 3.1. The FIS-S and FIS-A are embedded into NS-3 code.

This modified NS-3 code is the implementation of EELM-Fy model.

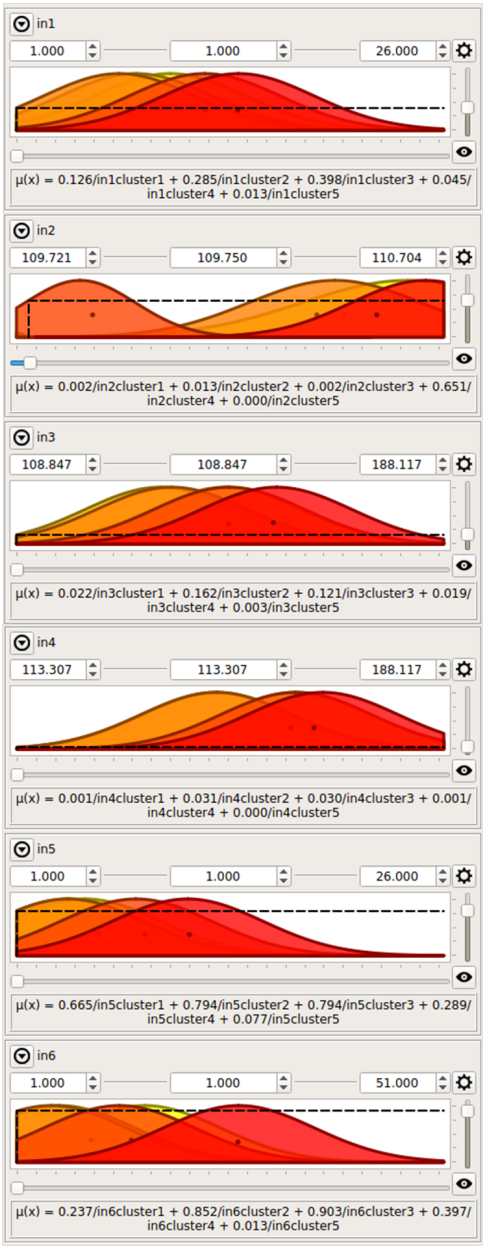


Figure 4.5: Input variables for FIS-A

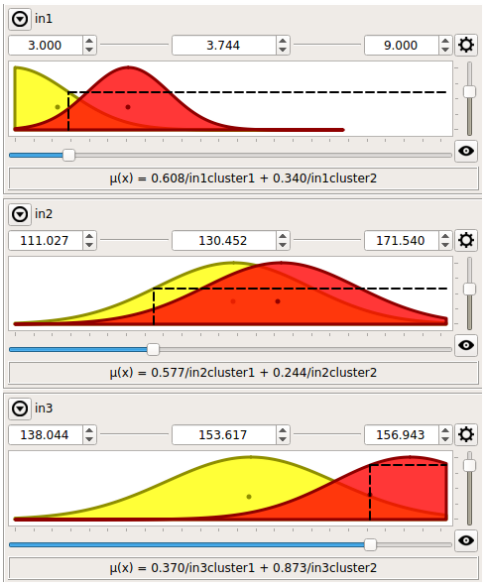


Figure 4.6: Input variables for FIS-S

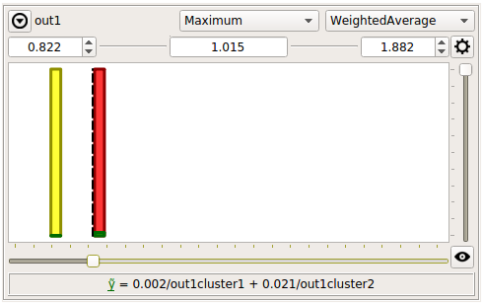


Figure 4.7: Output variable for FIS-S

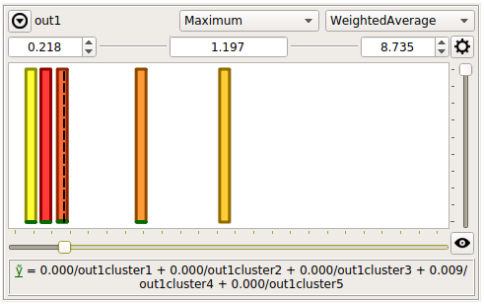


Figure 4.8: Output variable for FIS-A

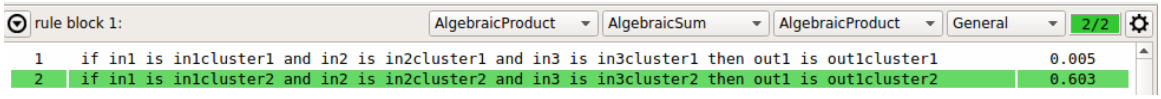


Figure 4.9: Rule block for FIS-S

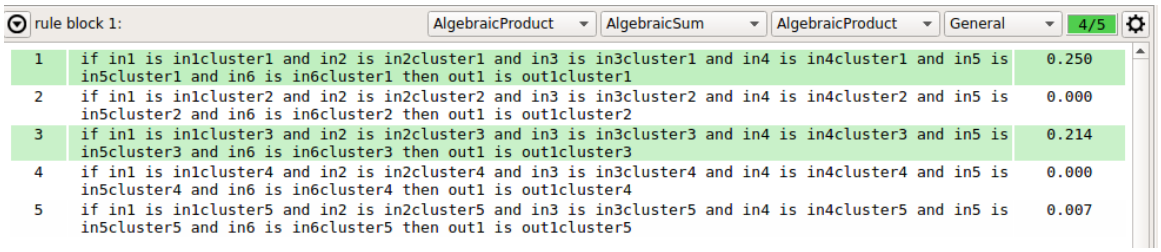


Figure 4.10: Rule block for FIS-A

4.2 Conclusion

We propose EELM-Fy because EELM-Fy does not need to decide the weights manually and subjectively. Therefore one can run EELM with random weights in the real environment and collect the data to train EELM-Fy, then get the trained EELM-Fy model, which fit that environment nearly perfectly, applying in the real environment.

The difference between EELM and EELM-Fy is that EELM-Fy uses two FISs to replace two utility functions in EELM. The steps of transforming EELM to EELM-Fy are described as follows:

1. Use EELM generate training data.
2. train two FISs by ANFIS Editor in Matlab
3. In NS-3, call Fuzzylite library to use the generated FISs from Matlab.

Chapter 5

Implementation

"Simple is better than a clever words touched my heart."

William Shakespeare

5.1 Simulation Assumptions

The assumptions for simulation are listed below:

1. All nodes are time-synchronized.
2. Nodes have knowledge about various cross-layer information, such as topology, connectivity, and residual battery status.
3. Sensor nodes are randomly deployed underwater while anchor nodes randomly float on the water surface.
4. Sensor nodes are aware of their depths.

5.2 Simulation Setting

We simulate 2 UASNs scenes called simulation 1 and simulation 2 described in Section 5.7. We use the concept of Bernoulli trial to attain simulation results. For each set of data, we run the simulation 1000 times with the same deployment and parameters, and calculate the average value as the result. For example, in simulation 1, in a $2500 \times 2500 \times 2500$ underwater area, we deploy 4 anchor nodes and 50 sensor nodes randomly, and we set a set of parameters (e.g. speed of current is 2 m/s). In this scene of simulation, we run EELM model and other localization models 1000 times respectively. Then we calculate the average values from 1000 times simulation results.

5.3 Packets Transmission Mechanism in UAN Module

Making on-field experiment to compare different topology control methods is very expensive and there is no commonly accepted standard to base on, thereby we use NS-3 simulator to simulate EELM and OLTC model. NS-3 simulator is a discrete-event network simulator targeted primarily for research and educational use [47]. All results will be discussed in Section 6. In NS-3 simulator, UAN module standardizes the process of modeling a variety of underwater network scenarios, therefore we use mainly UAN module in the simulation. The UAN Module consists of mainly three parts: the propagation, physical, and MAC module.

Figure 5.1 shows how one packet sent and received by the sensors in programming level. First, an object of class $UanNetDevice$ [48] calls the function $Send()$ to pass the packet to MAC layer, in function $Send()$, it will get an object of class $UanMacCw$ [49] and the object will call $Enqueue()$ to pass the packet to the physical layer. In function $Enqueue()$, it will get an object of class $UanPhyGen$ [50], then the object will call function $SendPcket()$ to pass the packet to a transducer, which is attached to the object of class $UanNetDevice$. In function $SendPcket()$, it will get an object of class $UanTransducerHd$ [51], then the object can call function $Transmit()$ to pass the packet to the transmission channel. In function $Transmit()$, it will get an object of class $UanChannel$ [52], then the object will call function $TxPacket()$ to transmit the packet. In function $TxPacket()$, it schedules the function $SendUp()$. In function $SendUp()$, it will get other objects of class $UanNetDevice$. Other objects of class $UanNetDevice$ will call function $SetReceiveCallback()$ to receive the packet.

In all classes we used, if there are parameters we changed, we will mention them in following sections, otherwise, they remain the default values. For all classes regarding sending and receiving we mentioned before, Figure 5.1 also shows the relations between superclass and subclass in Unified Modeling Language (UML) class diagram standard. For example, class $UanNetDevice$ is the subclass of $NetDevice$.

5.4 UAN Propagation Module

UAN module provides three propagation models. All the propagation models provide a power delay profile (PDP) information and pathloss between the source and receiver in dB re 1uPa.

a) Ideal Channel Model. It assumes there is no pathloss inside a cylindrical area with bounds set by attribute. The ideal channel model also assumes an impulse PDP.

b) Thorp Propagation Model. It assumes an impulse channel response, and the implement is similar to the implementation in ns2 [42]. The difference is that it uses center frequency (defined in UanTxMode [53]) in calculation.

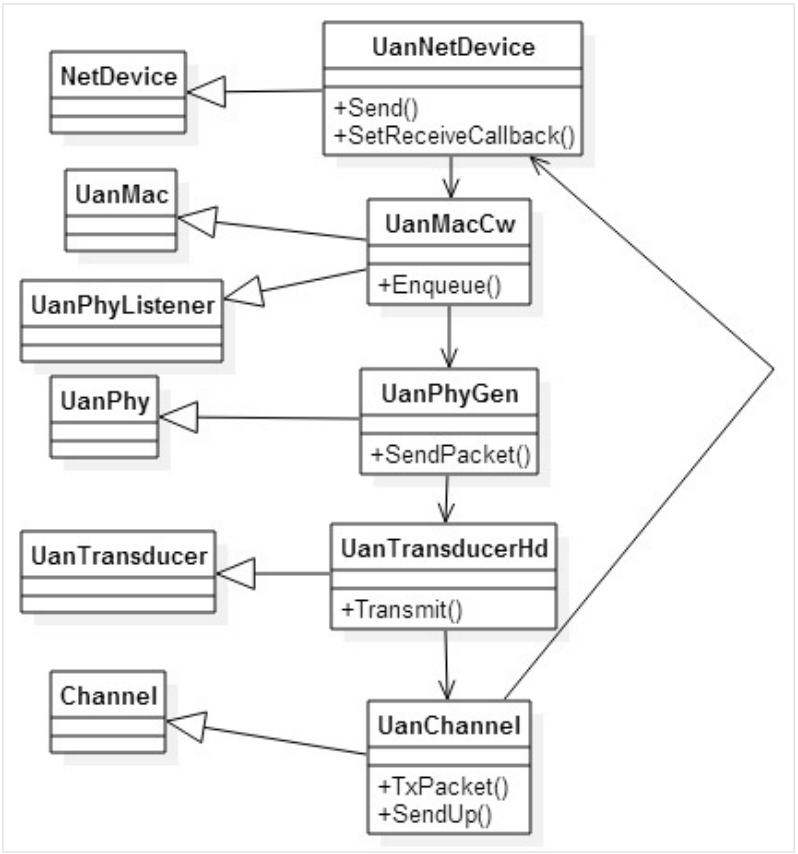


Figure 5.1: Sending and receiving packets mechanism in UAN model

c) Bellhop Propagation Model. It reads propagation information from a database. A configuration file describing the location, and resolution of the archived information must be supplied via attributes.

We use thorp propagation model because ideal channel model has no pathloss, and Bellhop propagation model is in developing, so it is not included in the newest NS-3 release.

The attenuation is calculated based on the spreading loss [54] and absorption loss. The attenuation is given as [55] [56] :

$$a = SpreadCoef \cdot 10 \cdot log_{10}(d) + (\frac{d}{1000}) \cdot T \tag{5.1}$$

Where a is the attenuation in dB, and $SpreadCoef$ is spreading coefficient which defines the geometry of the propagation (i.e., k = 1 is cylindrical, k = 2 is spherical, and k = 1.5 is practical spreading. Here we use the NS-3 default set $SpreadCoef = 1.5$. d is the distance, and T is the Thorp's approximation [57] for absorption loss calculated by Equation (5.2).

In thorp propagation model, we use center frequency to get the Thorp's approximation for the absorption loss in dB/Km by Equation (5.2) [55] [56].

$$T = \begin{cases} \frac{0.11 \cdot f^2}{1+f^2} + \frac{44 \cdot f^2}{4100+f^2} + 2.75 \cdot 10^{-4} \cdot f^2 + 0.003 & f \geq 0.4 \\ 0.002 + 0.11 \cdot \frac{f^2}{1+f^2} + 0.11 \cdot f^2 & f < 0.4 \end{cases} \tag{5.2}$$

Where T is the Thorp's approximation for absorption loss in dB/Km, and f is center frequency in kHz.

Note that article [56] gives the general formulas for attenuation. However, Equation (5.1) and (5.2) are slightly different from the general formulas. We use Equation (5.1) and (5.2) rather than general formula, because in NS-3 UAN model implements the attenuation with Equation (5.1) and (5.2) [42] (the source code can be found here [55]).

5.5 Physical Layer

In this layer, we use the generic physical class: UanPhyGen. The general responsibility of UanPhyGen is to handle packet acquisition, error determination, and forwarding of successful packets up to the MAC layer. The UanTxMode class describes how the physical class responds to packets. The attributes in UanTxMode is described in Table 5.1.

Class UanPhyGen needs the combination of Signal to Noise Ratio (SINR) and Packet Error Rate (PER) to determine successful reception of packets. There are several different PER and SINR models.

a) PER models

- Default PER model. It tests the packet against a threshold and assumes error (with probability 1) if the SINR is below the threshold or success if the SINR is above the threshold.
- Micromodem FH-FSK PER [7]. It calculates probability of error assuming a rate 1/2 convolutional code with constraint length 9 and a CRC check capable of correcting up to 1 bit error.

b) SINR models

- Default SINR Model. It assumes that all transmitted energy is captured at the receiver and that there is no ISI (InterSymbol Interference). Any received signal power from interferes acts as additional ambient noise.
- FH-FSK SINR Model. It is similar to the model proposed by [7].
- Frequency filtered SINR. It only considers interference if there is an overlap in frequency of the arriving packets as determined by UanTxMode, otherwise, it calculates SINR in the same way as the default model.

We use the default PER (UanPhyPerGenDefault) and SINR model (UanPhyPerUmodem) in the simulation. A crucial parameter, $TxPower$, which decides the transmission range, is adjusted in the physical layer. We get an object of UanPhyGen class to dynamically adjust $TxPower$ in dB by the function $SetTxPowerDb()$. In the simulation, we can calculate $TxPower$ value by following formula:

$$TxPower = a + NI \tag{5.3}$$

where a is attenuation calculated by Equation (5.1), and NI is the noise intensity.

Table 5.1: Physical attributes in class $UanTxMode$

Parameter	Value
Center frequency	22 Khz
Modulation technique	FSK
Data rate	500 Bps
Symbol rate	80 Sps
Bandwidth	4000 Hz

5.6 MAC Layer

We discuss MAC layer because the simulation results, such as energy consumption, localization error and localization delay, are directly influenced by MAC layer protocols. There are three MAC protocols are implemented in UAN model.

a) CW-MAC [58]. It uses a slotted contention window similar in nature to the IEEE 802.11 DCF. Nodes have a constant contention window measured in slot times (configured via $SetCw()$ function in UanMacCw). If the channel is sensed busy, then nodes backoff by randomly (uniform distribution) choose a slot to transmit in.

b) RC-MAC. It dynamically divides the available bandwidth into a data channel and a control channel. RTS/CTS handshaking is used and time is divided into cycles.

c) Simple ALOHA. No collision control at all, nodes transmit packet freely.

We use CW-MAC in MAC layer in simulation, because RC-MAC needs a gateway node which all network traffic is destined for, which does not fit our model and Simple ALOHA cannot prevent and reduce collisions. However, CW-MAC cannot prevent all collisions, because, in simulation, all nodes communicate by broadcasting. There are exposed node problem and hidden node problem [59]. We cannot use RTS/CTS mechanism [60], [61] to avoid the collisions, because underwater signal speed is too slow, RTS/CTS signal increases the network delay and brings down the throughput. Due to the delay and uncertainty of underwater communication, we cannot predict when the packet arrives and when RTS/CTS signal arrives. The target node may already turn off to sleep mode during the waiting, which may eventually cost more transmission channel resources.

Base on CW-MAC protocol, we schedule and control back-off time for anchor nodes, because, for localization model, the localization coverage is primary performance metric. CW-MAC protocol cannot prevent all collision of packets, and the packets from anchor nodes have a big impact on localization coverage. Therefore, we run CW-MAC protocol on both anchor and sensor node, and base on that, we schedule back-off time for anchor nodes. For example, at time 0, anchor node 0 sends its message, and at time 0.5, anchor node 1 sends its message. This time slot can be changed at any time considering the transmission range of the anchor node.

5.7 Nodes Deployment and Mobility Model

We set two simulation scenes to test the models, it reveals strengths and the drawbacks of models in different situations.

In first simulation scene, we randomly deploy 10-50 sensor nodes in a $2500 \times 2500 \times 2500$ (Length $\times$ Width $\times$ Deep) underwater area, 4 anchor nodes are randomly deployed on the surface of the same area. The speed of ocean current is $2m/s, 3m/s, 4m/s$. Every node moves randomly according to the ocean current. For example, if the coordinate of a sensor node is $(x_i, y_i, z_i, i = 1 \cdots n)$, and when the speed of ocean current is $2\ m/s$, the coordinate of sensor nodes will be $(x_i \pm 2, y_i \pm 2, z_i \pm 2)$ at next second. By iterating this process the ocean current is simulated. The anchor nodes also influenced by the ocean current but they stay on the water surface, so the coordinate z remains the

same. For example, if the coordinate of an anchor node is $(x_j, y_j, z_j, j = 1 \cdots n)$ at second 0, and when the speed of ocean current is 2 m/s, the coordinate of sensor nodes will be $(x_j \pm 2, y_j \pm 2, z_j)$ at second 1. The max transmission range is $Max\_Range = \sqrt{2500^2 + 2500^2 + 2500^2}$, and the initial transmission range is $Initial\_Range = Max\_Range/2$.

In the second simulation scene, we randomly deploy 10-50 sensor nodes in a $10000 \times 10000 \times 2500$ (Length $\times$ Width $\times$ Deep) underwater area, 4-20 anchor nodes are randomly deployed on the surface of the same area. The max transmission range is $Max\_Range = 5000$, and the initial transmission range is $Initial\_Range = Max\_Range/2$.

In summary, the simulation scenes can be found in Table 5.2.

Table 5.2: Summary of simulation scenes

Simulation	Sensor Type	Nodes Number	Simulation Area	max_rang	min_range	Ocean Speed
1	anchor node sensor node	4 {10, 20, 30, 40, 50}	$2500^3 m^3$	$\sqrt{2500^2 \times 3}$	$\frac{max\_range}{2}$	$\{2, 3, 4m/s\}$
2.1	anchor node sensor node	20 {10, 20, 30, 40, 50}	$10000^2 m^2 \times 2500m$	5000m	2500m	$\{2m/s\}$
2.2	anchor node sensor node	{4, 8, 12, 16, 20} 50	$10000^2 m^2 \times 2500m$	5000m	2500m	$\{2m/s\}$

NS-3 has many build-in mobility models. The deployment and the mobility model can be replaced anytime.

Using random deployment may lead to some extreme situations, for example, all anchor nodes are deployed on the top right corner, and all sensor nodes are deployed around the button left corner. Obviously, in this situation, the localization coverage is 0, because the transmission range is less than the distance between anchor nodes and sensor nodes. However, the extreme situation also exists in real world. In the simulation, we simulate the process for 1000 times, then take the average results to reduce impacts from extreme situations.

5.8 Energy Consumption Model

We use class $BasicEnergySourceHelper$ to create energy source object for every node. We use class $AcousticModemEnergyModelHelper$ to install the energy model in every node. The default power consumption values are set according to the article [7]. But in simulation, the initial energy setting is shown in Table 5.3, in which the initial transmission power is different according to different simulation scenes described in 5.7. The transmission power will change during the topology control process.

The relation between sound pressure level (SPL) in dB and sound power in watt is calculated in

following formula [62].

$$L_W = 10log_{10}(\frac{P}{P_0}) \tag{5.4}$$

where L_W is SPL in dB, P is sound power in watt, P_0 is reference sound power in watt. We use this formula because we control transmission range by controlling SPL through modifying a parameter $TxPower$ in class $UanPhyGen$, and the energy model cannot detect the changes of $TxPower$ automatically. The class $AcousticModemEnergyModel$ records the energy consumption automatically according to the state change of the nodes. However, it cannot detect transmission power change in the physical layer (in dB). For example when we change the parameter $TxPower$ (in dB) in physical layer ($TxPowers$ will change transmission range eventually), class $AcousticModemEnergyModel$ still records the transmission energy consumption (calculated by power(in watt) $\times$ time) by initial transmission power in watt (57.9 w or 104.7 w). So every time we modify the SPL in physical layer to control the transmission range, we need to call the method $SetTxPowerW()$ in class $AcousticModemEnergyModel$ again to reset transmission power in watt, then class $AcousticModemEnergyModel$ can record energy consumption correctly.

Table 5.3: Attributes in class $AcousticModemEnergyModel$

Modem State Power Consumption	
Initial transmission power	57.9 w or 104.7 w
Receiving	0.1 w
Idle	0.1 w
Sleep	0.0001 w
Initial energy	2000 J

5.9 Message Formats

Table 5.4 denotes the first broadcast format, in which Flag will help the receiver recognizes that this is a “one-hop" message (Flag=1). Node Type will help the receiver recognizes the node type of the sender (0 denotes anchor node, 1 denotes unknown node). ID denotes the sender‘s ID $(1, 2, \cdots, n)$, where n is the number of senders. Time records the sending time of this packet, which helps the receiver calculate the distance between the sender and itself.

Table 5.4: First broadcast format

Flag	Node Type	ID	Time

Table 5.5 denotes the second broadcast format, in which Flag will help the receiver recognizes that this is a “two-hop" message (Flag=2). Node Type will help the receiver recognizes the node

type of the sender. ID denotes the sender's ID $(1, 2, \cdots, n)$. Time records the sending time of this packet. Neighbor Table contains IDs and distances of sender's neighbors. N denotes the number of sender's neighbors.

Table 5.5: Second broadcast format

Flag	Node Type	ID	Time	Neighbor Table	N

Table 5.6 denotes the third broadcast format, in which Flag will help the receiver recognizes that this is a "Request" message (Flag=3). Node Type will help the receiver recognizes the node type of the sender. ID denotes the sender's ID $(1, 2, \cdots, n)$. Time records the sending time of this packet. n_{req} denotes how many "Reply" messages still needed for the sender to localize itself. P denotes the transmission power selected by sender's payoff.

Table 5.6: Request (third broadcast format)

Flag	Node Type	ID	Time	n_{req}	P

Table 5.7 denotes the fourth broadcast format, in which Flag will help the receiver recognizes that this is a "Reply" message (Flag=4). Node Type will help the receiver recognizes the node type of the sender. ID denotes the sender's ID $(1, 2, \cdots, n)$. Time records the sending time of this packet. Location contains the coordinate of the sender at the sending time.

Table 5.7: Reply (forth broadcast format)

Flag	Node Type	ID	Time	Location

5.10 Conclusion

In this chapter, we describe all necessary factors to realize EELM in practical, which include simulation assumptions, settings, nodes deployment, mobility model, and message formats. In addition, we illustrate packets transmission mechanism in UAN Module in order to help understand the programming in NS-3 UAN model for EELM. Besides we describe all basic programming settings used in EELM including propagation module, energy consumption model, and protocols in physical and MAC layer. With all parameters and settings described in this chapter, one can reproduce the simulation results.

Chapter 6

Analysis of Simulation Results

"All of the reasons must come from observation and experiments."

Galileo Galilei

In this chapter we compare simulation results from following five models:

1. EELM: Energy Efficient Localization Model (EELM) is one of the topology control models proposed in this paper. It uses Stackelberg game to control the transmission power for anchor and sensor nodes in topology control. The main purpose of EELM is saving energy for sensor nodes. The detail is described in Section 3.

2. EELM-Min: This is a reference model modified from EELM. It shows the upper bound of the simulation results. In this model, the topology control process is as same as EELM, but all nodes use minimum transmission power to send the message without consideration of neighbor information and any gaming process.

3. EELM-Max: This is a reference model modified from EELM. It shows the lower bound of the simulation results. In this model, the topology control process is as same as EELM, but all nodes use maximum transmission power to send the message without consideration of neighbor information and any gaming process.

4. EELM-Fy: Energy Efficient Localization Model using Fuzzy logic control system (EELM-Fy) is one of the topology control models proposed in this paper. Base on EELM, it uses fuzzy logic control in gaming process to calculate the payoffs for anchor and sensor nodes, and the training data is gained from EELM. The detail is described in Section 4.

5. OLTC: Opportunistic Localization by Topology Control (OLTC) is a localization scheme proposed by [6]. It uses Stackelberg game theory in topology control for controlling the transmission power in anchor and sensor nodes.

6.1 Performance Metrics

Communication cost, coverage, time, and accuracy are common performance metrics for localization algorithm [4]. We take following 6 performance metrics to validate EELM model against the previous state of the art. Suppose N_{an} is anchor node set, N_{sn} is sensor node set, and N_{sn_l} is localized sensor node set. Obviously $N_{sn\_l} \subseteq N_{sn}$. $|N_{sn}|$ denotes the number of sensor nodes in N_{sn}. $|N_{an}|$ denotes the number of anchor nodes in N_{an}. $|N_{sn\_l}|$ denotes the number of sensor nodes in N_{sn_l}.

1) Localization coverage. It is the ratio of the number of sensor nodes which can localize itself to the total number of sensor nodes, that is $Coverage = \frac{|N_{sn\_l}|}{|N_{sn}|}$.

2) Average energy consumption per sensor node. It is the ratio of the number of sensor nodes to the total energy consumed by sensor nodes. It is denoted by $E_{sn}^{avg} = \frac{\sum_{i=1}^{N_{sn}} E_i}{|N_{sn}|}$, in which E_i is the energy consumption of sensor node i in topology control. $E_i = E_{request}^i - E_{reply}^i$, in which $E_{request}^i$ is residual energy of sensor node i before sending 'Request' message, E_{reply}^i is residual energy of sensor node i after receiving 'Reply' message.

3) Average energy consumption per anchor node. It is the ratio of the number of anchor nodes to the total energy consumed by anchor nodes. It is denoted by $E_{an}^{avg} = \frac{\sum_{j=1}^{N_{an}} E_j}{|N_{an}|}$, in which E_j is the energy consumption of anchor node j in topology control. $E_j = E_{wake\_up}^j - E_{reply}^j$, in which $E_{wake\_up}^j$ is residual energy of anchor node i before sending first wake up message, E_{reply}^j is residual energy of anchor node j after sending 'Reply' message.

4) Average energy consumption per node. It is the ratio of the total energy consumption of all nodes to the number of total nodes, which is calculated as $E_{total}^{avg} = \frac{\sum_{i=1}^{|N_{sn}|} E_i + \sum_{j=1}^{|N_{an}|} E_j}{|N_{sn}|+|N_{an}|}$, where E_i and E_j denote the energy consumption of sensor nodes i and anchor nodes j respectively.

5) Average localization error. It is calculated by using the following formula: $\frac{\sum_{i=1}^{|N_{sn\_l}|} \sqrt{(x_i-x_i')^2+(y_i-y_i')^2+(z_i-z_i')^2}}{|N_{sn\_l}|}$, where for any localized sensor node i, (x_i, y_i, z_i) and (x_i', y_i', z_i') denote the real and the estimated locations, respectively.

6) Average localization delay. It is calculated by $D_{avg} = \frac{\sum_{i=1}^{|N_{sn\_l}|} D_i}{|N_{sn\_l}|}$, where D_i is the time from an sensor node broadcasting a 'Request' message to the time the node localizing itself.

6.2 Localization Coverage

Figure 6.1, 6.4, and 6.3 show average localization coverage (percentage of localized sensor nodes) along with changes in the number of sensor or anchor nodes in two simulation scenes. The higher the average localization coverage the better the UASNs localization model is.

In Figure 6.1, the number of anchor node is 4. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. The simulation scene is simulation scene 1 described in Section 5.7. Average localization coverage in EELM-Min stays lowest compared with other models, because it always uses the minimum transmission power to send messages without consideration of neighbor information, thus fewer sensor nodes can receive enough information to localize themselves. In the contrast, average localization coverage in EELM-Max stays highest compared with other models, especially when the number of sensor nodes is low. Average localization coverages in EELM, EELM-Fy, and OLTC are between that in EELM-Min and EELM-Max, in which average localization coverages in EELM and EELM-Fy are slightly higher than that in OLTC (in the order of 1-3%) when the number of sensor nodes is low (e.g. 10 and 20). Along with the raising of the number of sensor nodes from 10 and 20 to 40 and 50, average localization coverages in EELM, EELM-Fy, EELM-Max, and OLTC are very close (in the order of 0.1%). In this data, we can see that the number of sensor nodes is the number of samples. The higher the number of samples, the more stable the result is. In summary, average localization coverage in EELM, EELM-Fy, EELM-Max, and OLTC are very close. And the trend of average localization coverage is steady, because the change in the number of samples does not influence the nature of this simulation.

In Figure 6.4, the number of sensor nodes is 50. The number of anchor nodes varies from 4 to 20. The speed of current is 2 m/s. The simulation scene is simulation scene 2 described in Section 5.7. As the number of samples is steady (50 sensor nodes), the results are more stable. Average localization coverage in EELM-Min still stays lowest compared with other models, because it always uses the minimum transmission power to send messages without consideration of neighbor information, thus fewer sensor nodes can receive enough information to localize themselves. Average localization coverages in EELM, EELM-Fy, EELM-Max, and OLTC are very close (in the order of 0.1%). There are up trends in average localization coverage for all models (27%-98% in EELM, EELM-Fy, EELM-Max and OLTC, and 1%-%41 in EELM-Min), because when the number of anchor nodes raises, more sensor nodes can receive enough information to localize themselves. All trends in simulation 2 are more stable than that in simulation 1. because the number of samples is steady at 50.

In Figure 6.3, the number of anchor node is 20. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. The simulation scene is simulation scene 2 described in Section 5.7. The trends and numerical ratio relations are similar to Figure 6.1, and when the number of samples raises, results of all models tend to steady. This is a control group corresponding to Figure 6.4. We can see that no matter how the number of anchor nodes or sensor nodes varies, average localization coverages in EELM, EELM-Fy, and OLTC are close (less than 2%).

Even if EELM and EELM-Fy do not have huge strength in average localization coverage comparing with OLTC, we still discuss and visualize it. Because average localization coverage is an important performance metrics in UASNs localization models, and we want to show that the performance of EELM and EELM-Fy in this area are slightly better or at least not worse than OLTC.And base on

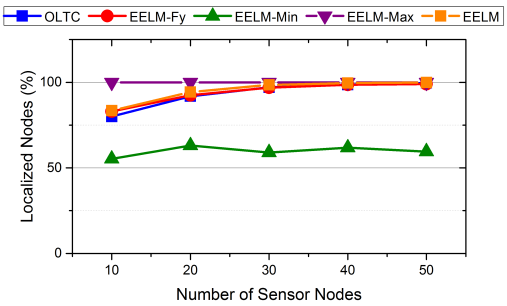


Figure 6.1: Average localization coverage from simulation 1

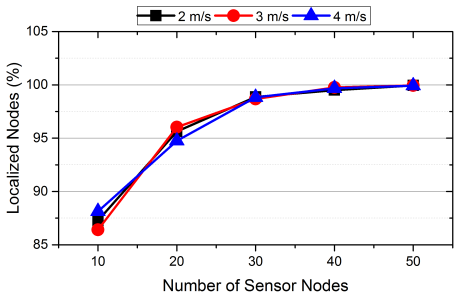


Figure 6.2: Average localization coverage with different speed of current

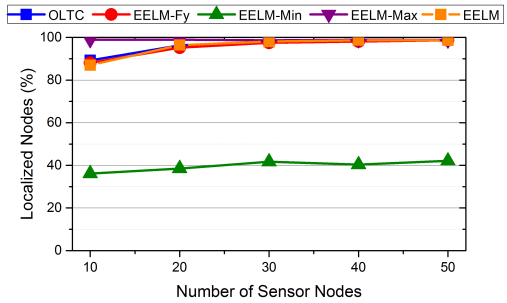


Figure 6.3: Average localization coverage from simulation 2 with different number of sensor nodes

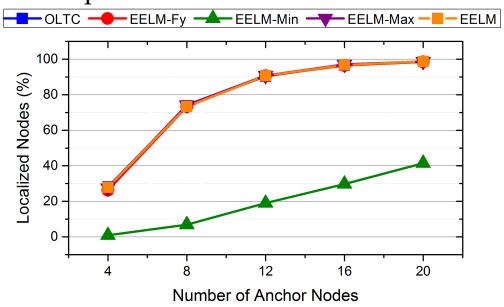


Figure 6.4: Average localization coverage from simulation 2 with different number of anchor nodes

equal performance in this metric, in energy efficiency described in Section 6.3 EELM and EELM-Fy are better than OLTC.

6.3 Average Energy Consumption

Figure 6.5, 6.6, and 6.7 show average energy consumption per sensor node along with changes in the number of sensor or anchor nodes in two simulation scenes. the time for calculating energy consumption is recorded from sending localization request message to receiving enough information for localization. The lower the average energy consumption per sensor node the better the UASNs localization model is.

Figure 6.5 shows the results of simulation scene 1 described in Section 5.7. The number of anchor node is 4. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. Average energy consumption per sensor node in EELM-Min and EELM-Max is the lowest and the highest respectively, because they always use minimum or maximum transmission power to send messages without considering the neighbor information. OLTC also has highest energy consumption in sensor nodes, which is like EELM-Max, but the difference is that sensor nodes

select the highest transmission power to send the message based on gaming process. Average energy consumptions per sensor node in EELM and EELM-Fy are between that in EELM-Min and EELM-Max. Energy consumptions in EELM and EELM-Fy are 3.5 times (500 J in value) lower than that in OLTC. There is no obvious trend in energy consumption when the number of sensor nodes raises. And as the number of sensor nodes represents the number of samples, the more sensor nodes there are the more stable the result is.

Figure 6.6 shows the results of simulation scene 2 described in Section 5.7. The number of sensor nodes is 50. The number of anchor nodes varies from 4 to 20. The speed of current is 2 m/s. Average energy consumption per sensor node in EELM-Min and EELM-Max is the lowest and the highest respectively. OLTC also has highest energy consumption in sensor nodes, which is like EELM-Max, but the difference is that sensor nodes select the highest transmission power to send the message based on gaming process. Average energy consumptions per sensor node in EELM and EELM-Fy between that in EELM-Min and EELM-Max, in which energy consumptions in EELM and EELM-Fy are 4.77-48.7% (83-892 J in value) lower than that in OLTC. There is a down trend of energy consumption in EELM and EELM-Fy when the number of anchor nodes raises, because when the number of anchor nodes raises, sensor nodes can choose smaller transmission ranges to broadcast localization requests, thus they use less transmission power to transmit packets, then energy is saved eventually. The trends of energy consumption in EELM-Max and OLTC are close, and both of them consume more energy in sensor nodes comparing other models. Because, in both of them, sensor nodes use maximum transmission power to send messages, even though OLTC use game theory to decide transmission power.

In Figure 6.7, the number of anchor node is 20. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. The simulation scene is simulation scene 2 described in Section 5.7. The trends and numerical ratio relations are the similar to Figure 6.5. This is a control group corresponding to Figure 6.6. Average energy consumption per sensor node in EELM and EELM-Fy are 48.24-52.98%(882-968 J in value) lower than that in OLTC. Combine Figure 6.6 and 6.7, we can see that no matter how the number of anchor nodes or sensor nodes varies, average energy consumptions per sensor node in EELM and EELM-Fy are lower than that in OLTC.

Figure 6.8, 6.9, and 6.10 show average energy consumption per anchor node along with changes in the number of sensor or anchor nodes in two simulation scenes. the time for calculating energy consumption is recorded from sending the wake-up message to after sending the reply message. The lower the average energy consumption per anchor node the better the UASNs localization model is.

Figure 6.8 shows the results of simulation scene 1 described in Section 5.7. The number of anchor node is 4. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. Average energy consumption per anchor node in EELM-Min and EELM-Max is the lowest and the highest respectively, because they always use minimum or maximum transmission power

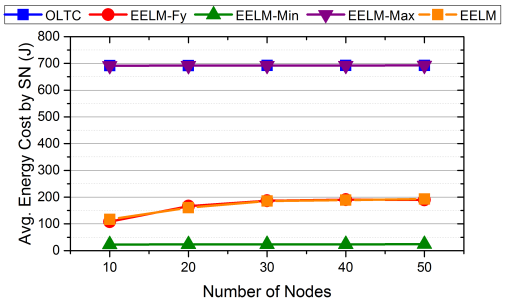


Figure 6.5: Average energy consumption per sensor node from simulation 1

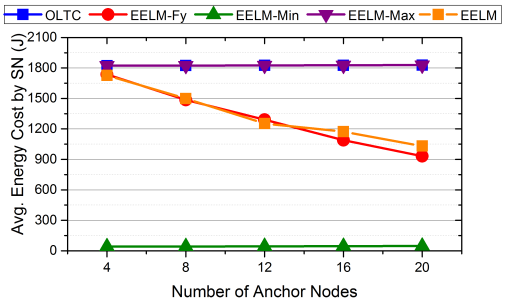


Figure 6.6: Average energy consumption per sensor node from simulation 2 with different number of anchor nodes

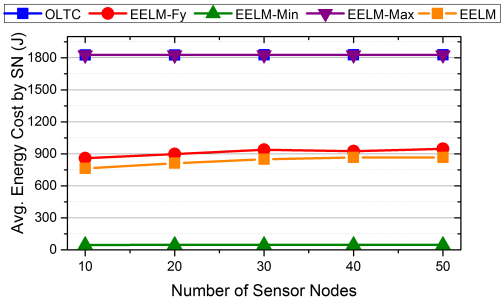


Figure 6.7: Average energy consumption per sensor node from simulation 2 with different number of sensor nodes

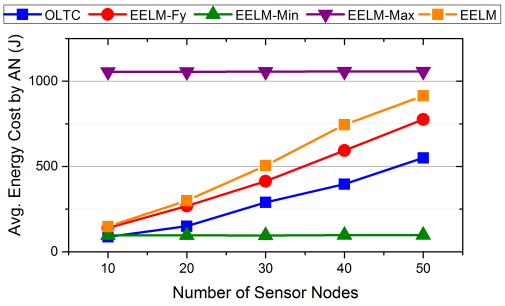


Figure 6.8: Average energy consumption per anchor node from simulation 1

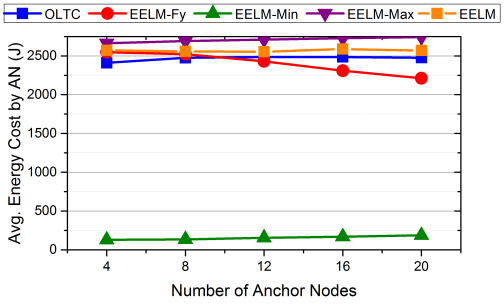


Figure 6.9: Average energy consumption per anchor node from simulation 2 with different number of anchor nodes

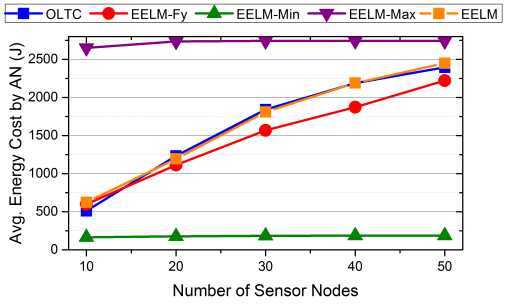


Figure 6.10: Average energy consumption per anchor node from simulation 2 with different number of sensor nodes

to send messages without considering the neighbor and request information. Average energy consumptions per anchor node in EELM, EELM-Fy, and OLTC are between that in EELM-Min and EELM-Max, in which energy consumptions in EELM and EELM-Fy are 66-70% (61- 364 J in value) higher than that in OLTC. In this performance metric, OLTC is better than EELM, because the purpose of EELM is saving more energy in sensor nodes at the cost of more energy consumption in anchor nodes. In the contrast, OLTC considers more in saving energy in anchor nodes. There are up trends in average energy consumption per anchor node in EELM, EELM-Fy, and OLTC, because when the number of sensor nodes raises, anchor nodes receive more localization requests from sensor nodes, so they raise their transmission power, and eventually the energy consumption of anchor nodes raises.

Figure 6.9 shows the results of simulation scene 2 described in Section 5.7. The number of sensor nodes is 50. The number of anchor nodes varies from 4 to 20. The speed of current is 2 m/s. Average energy consumption per anchor node in EELM-Min and EELM-Max is the lowest and the highest respectively, because they always use minimum or maximum transmission power to send messages without considering the neighbor and request information. Average energy consumptions per anchor node in EELM, EELM-Fy, and OLTC are between that in EELM-Min and EELM-Max, in which energy consumptions in EELM-Fy is 5.68-10.6%(137-264J in value) lower than that in EELM and OLTC. In this simulation scene, even though EELM-Fy is trained from EELM which consumes more energy in anchor nodes in order to save energy in sensor nodes, it has better performance in saving energy in anchor nodes. The data only slight fluctuation in this simulation scene, because the number of sensor nodes is steady at 50. In anchor nodes, the localization requests received are always stable.

In Figure 6.10, the number of anchor node is 20. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. The simulation scene is simulation scene 2 described in Section 5.7.The trends are similar to Figure 6.8. This is a control group corresponding to Figure 6.9. EELM-Fy consumes 7.2-9.74% (120-174 J in value)less energy in anchor nodes than OLTC when the number of sensor nodes raises, even though EELM-Fy is trained from EELM.

Figure 6.11, 6.14, and 6.13 show average energy consumption per node along with changes in the number of sensor or anchor nodes in two simulation scenes. The way to calculate this value is described in Section 6.1. The lower the average energy consumption per node the better the UASNs localization model is.

Figure 6.11 shows the results of simulation scene 1 described in Section 5.7. The number of anchor node is 4. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. Average energy consumption per node in EELM-Min and EELM-Max is the lowest and the highest respectively, because they always use minimum or maximum transmission power to send messages without considering the neighbor information. Average energy consumptions per sensor node in EELM, EELM-Fy, and OLTC are between that in EELM-Min and EELM-Max, in which

energy consumptions in EELM and EELM-Fy are 63.8-66.6%(345-435 J in value) lower than that in OLTC. There is no obvious trend in energy consumption when the number of sensor nodes raises. And as the number of sensor nodes represents the number of samples, the more sensor nodes there are the more stable the result is. There are up trends in average energy consumption per node in EELM, EELM-Fy, and OLTC, the trends are the composite results from Figure 6.5 and 6.8. In Figure 6.5 the trend is stable, and in Figure 6.8 there is an up trend. Therefore the composite trend is an up trend.

Figure 6.14 shows the results of simulation scene 2 described in Section 5.7. The number of sensor nodes is 50. The number of anchor nodes varies from 4 to 20. The speed of current is 2 m/s. Average energy consumption per node in EELM-Min and EELM-Max is the lowest and the highest respectively, because they always use minimum or maximum transmission power to send messages without considering the neighbor and request information. Average energy consumptions per node in EELM, EELM-Fy, and OLTC are between that in EELM-Min and EELM-Max, in which energy consumptions in EELM and EELM-Fy is 1.33-26.5%(25-535 J in value) lower than that in OLTC. There are down trends in average energy consumption per node in EELM and EELM-Fy, and there are up trends in OLTC and EELM-Max. The trends are the composite results from Figure 6.6 and 6.9.

In Figure 6.13, the number of anchor node is 20. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. The simulation scene is simulation scene 2 described in Section 5.7. The trends are the composite results from Figure 6.7 and 6.10. This is a control group corresponding to Figure 6.14. Average energy consumption per node in EELM and EELM-Fy are 25.07-33.7% (238-671 J in value) lower than that in OLTC. Combine Figure 6.14 and 6.13, we can see that no matter how the number of anchor nodes or sensor nodes varies, average energy consumptions per node in EELM and EELM-Fy are lower than that in OLTC.

In summary of energy consumption in 5 models. EELM-Min and EELM-Max are always the lower and upper bounds. EELM and EELM-Fy consume more energy than OLTC for anchor nodes. However, EELM and EELM-Fy save more energy than OLTC for sensor nodes. And in average energy consumption per node, EELM and EELM-Fy are better than OLTC, which means EELM and EELM-Fy save more energy than OLTC in total. EELM and EELM-Fy are suitable for the situation where sensor nodes carry less energy and cannot recharge itself, and the anchor nodes have more energy source or can recharge itself (e.g. solar powered buoy or ship), while OLTC is suitable for the situation where the anchor nodes need to save energy. According to this analysis, we propose a future work idea here, which combines EELM and OLTC model. It is described in Section 7.2.

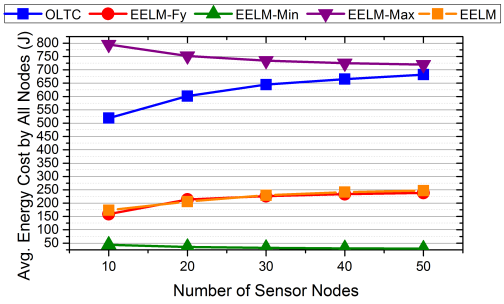


Figure 6.11: Average energy consumption per node from simulation 1

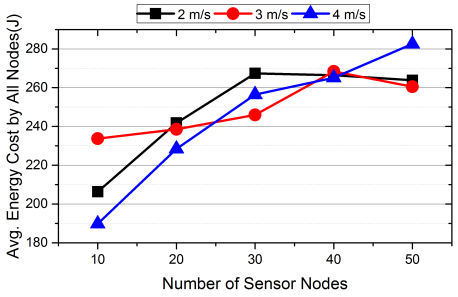


Figure 6.12: Average node energy consumption with different speed of current

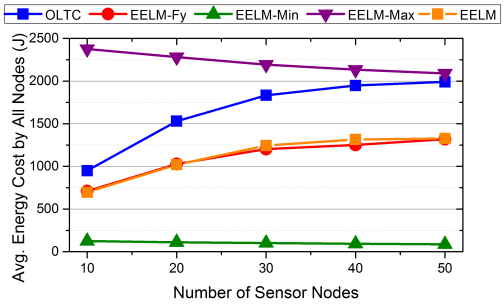


Figure 6.13: Average energy consumption per node from simulation 2 with different number of sensor nodes

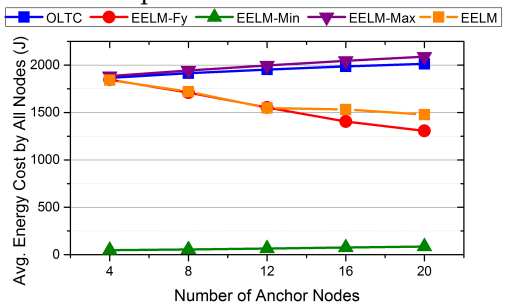


Figure 6.14: Average energy consumption per node from simulation 2 with different number of anchor nodes

6.4 Average Localization Delay

Figure 6.15, 6.18, and 6.17 show average localization delay (the time elapsed between sending a localization request to receiving enough information for localization) along with changes of the number of sensor or anchor nodes in two simulation scenes. The lower the average localization delay the better the UASNs localization model is.

In Figure 6.15, the number of anchor node is 4. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. The simulation scene is simulation scene 1 described in Section 5.7. Average localization delay in EELM-Min stays lowest compared with other models, because it always uses the minimum transmission power to send messages without consideration of neighbor information, thus the transmission range and delay become shorter. In the contrast, average localization delay in EELM-Max stays highest compared with other models, because it always uses the maximum transmission power to send messages without consideration of neighbor information, thus the transmission range and delay become longer. Average localization delays in EELM, EELM-Fy, and OLTC are between that in EELM-Min and EELM-Max, in which average localization delays in EELM, EELM-Fy, and OLTC are very close (0.1-0.4 m). There are up trends in average localization delay (6.22-6.5 s in EELM-Min and 6.55-6.9 s in EELM, EELM-Fy, and OLTC) when the number of sensor nodes raises, because the more nodes there are, the more back-off time is taken between sensor nodes in MAC layer (in Section 5.6).

In Figure 6.18, the number of sensor nodes is 50. The number of anchor nodes varies from 4 to 20. The speed of current is 2 m/s. The simulation scene is simulation scene 2 described in Section 5.7. Average localization delay in EELM-Min stays lowest compared with other models, because it always uses the minimum transmission power to send messages without consideration of neighbor information, thus the transmission range and delay become shorter. Average localization delays in EELM, EELM-Fy, OLTC, and EELM-Max are very close (0.1-0.2 m). There are up trends in average localization delay (5-23 s in EELM-Min and 6-31 s in EELM, EELM-Fy, EELM-Max, and OLTC) when the number of anchor nodes raises, because the more nodes there are, the more back-off time is taken between anchor nodes in MAC layer (in Section 5.6).

In Figure 6.17, the number of anchor node is 20. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. The simulation scene is simulation scene 2 described in Section 5.7. The trends and numerical ratio relations are similar to Figure 6.15. This is a control group corresponding to Figure 6.18. Combine Figure 6.18 and 6.17, we can see that no matter how the number of anchor nodes or sensor nodes varies, average localization delay in EELM, EELM-Fy, and OLTC are close (less than 3%).

Even if EELM and EELM-Fy do not have huge strength in average localization delay comparing with OLTC, we still discuss and visualize it. Because average localization delay is an important performance metrics in UASNs localization models, and we want to show that the performance of

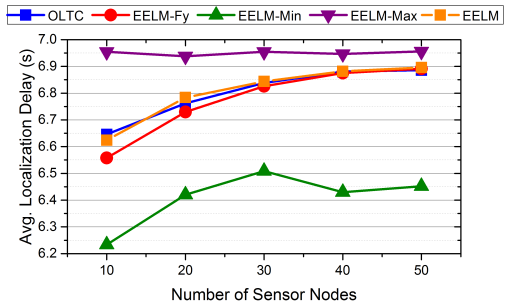


Figure 6.15: Average localization delay per node from simulation 1

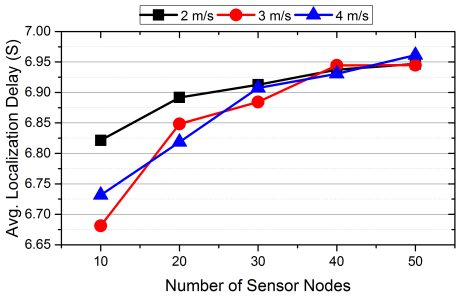


Figure 6.16: Average localization delay with different speed of current

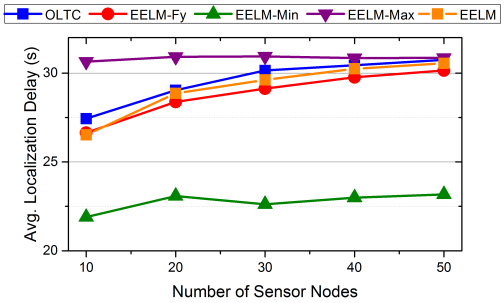


Figure 6.17: Average localization delay per node from simulation 2 with different number of sensor nodes

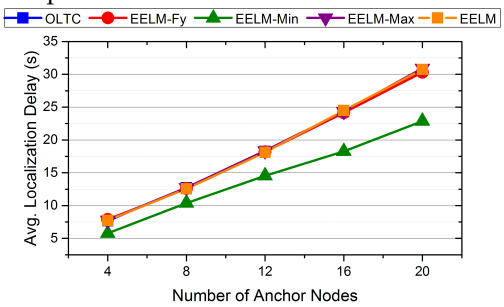


Figure 6.18: Average localization delay per node from simulation 2 with different number of anchor nodes

EELM and EELM-Fy in this area are slightly better or at least not worse than OLTC.And base on equal performance in this metric, in energy efficiency described in Section 6.3 EELM and EELM-Fy are better than OLTC.

6.5 Average Localization Error

Figure 6.19, 6.22, and 6.21 show average localization error (the Euclidean distance between real position and estimated position) along with changes in the number of sensor or anchor nodes in two simulation scenes. The lower the average localization error the better the UASNs localization model is.

In Figure 6.19, the number of anchor node is 4. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. The simulation scene is simulation scene 1 described in Section 5.7. Average localizations error in all models fluctuates slightly (3.15-3.35 m). The differences are small (0.1-0.2 m) considering the simulation scene (a $2500 \times 2500 \times 2500m$ area), so we consider all models have very close performances in this area. And there is no obvious trend in average localization error.

In Figure 6.22, the number of sensor nodes is 50. The number of anchor nodes varies from 4 to 20. The speed of current is 2 m/s. The simulation scene is simulation scene 2 described in Section 5.7. Average localizations error in all models fluctuates slightly (3.15-4.2 m). The differences are small (0.1-1.0 m) considering the simulation scene (a $10000 \times 10000 \times 2500m$ area), so we consider all models have very close performances in this area. And there is no obvious trend in average localization error.

In Figure 6.21, the number of anchor node is 20. The number of sensor nodes varies from 10 to 50. The speed of current is 2 m/s. The simulation scene is simulation scene 2 described in Section 5.7. The trends and numerical ratio relations are similar to Figure 6.19. This is a control group corresponding to Figure 6.22. Combine Figure 6.22 and 6.21, we can see that no matter which number of anchor nodes or sensor nodes varies, average localization error in EELM, EELM-Fy, and OLTC are close (less than 6% or 0.4 m in value).

Even if EELM and EELM-Fy do not have huge strength in average localization error comparing with OLTC, we still discuss and visualize it. Because average localization error is an important performance metrics in UASNs localization models, and we want to show that the performance of EELM and EELM-Fy in this area are slightly better or at least not worse than OLTC. And base on equal performance in this metric, in energy efficiency described in Section 6.3 EELM and EELM-Fy are better than OLTC.

6.6 Environmental Influences

Base on simulation scene 1 described in Section 5.7, we change the speed of current from 2 m/s to 4 m/s to observe how much influences the environment brings to the simulation results. The number of anchor node is 4. The number of sensor nodes varies from 10 to 50. The model is EELM.

Figure 6.2 shows average localization coverage with different speed of current. The data trend is consistent with Figure 6.1. When the speed of current varies, localization coverage varies slightly (1-2%) only when the number of samples is low (10 or 20 sensor nodes). When the number of sample raises, localization coverage has almost no change (0.01-0.02%) when the speed of current changes.

Figure 6.12 shows average node energy consumption with different speed of current. The data trend is consistent with Figure 6.11. When the speed of current varies, average node energy consumption varies. But there is no pattern, which means there is no direct connection between energy consumption and the speed of current.

Figure 6.16 shows average localization delay with different speed of current. The data trend is consistent with Figure 6.15. When the speed of current varies, average localization delay varies slightly (1.47-2.25%) only when the number of samples is low (10 or 20 sensor nodes). When the

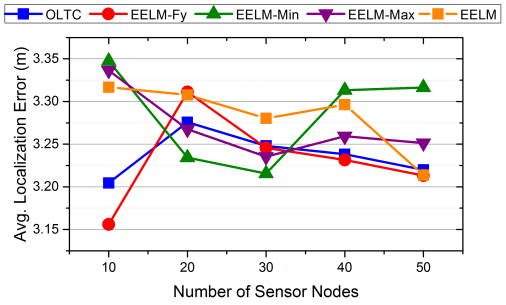


Figure 6.19: Average localization error per node from simulation 1

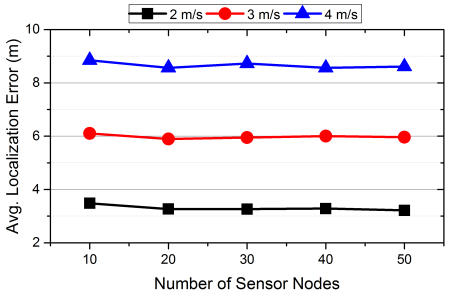


Figure 6.20: Average localization error with different speed of current

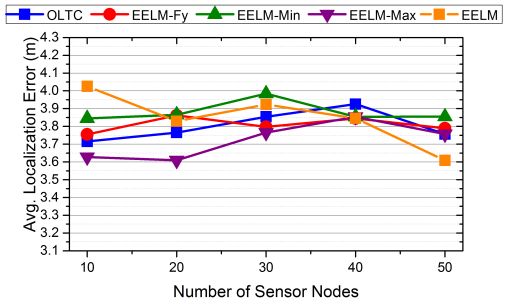


Figure 6.21: Average localization error per node from simulation 2 with different number of sensor nodes

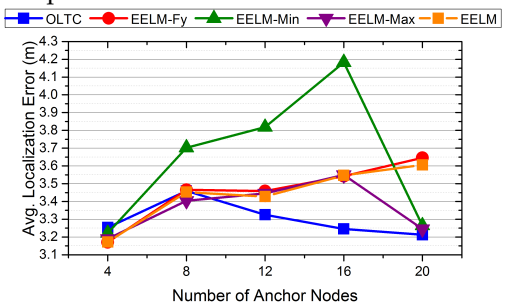


Figure 6.22: Average localization error per node from simulation 2 with different number of anchor nodes

number of sample raises, average localization delay has almost no change (0.01-0.03%) when the speed of current changes.

Figure 6.20 shows average localization error with different speed of current. The data trend is consistent with Figure 6.19. When the speed of current raises from 2 m/s to 4 m/s, average localization error raises from 3.2-3.4 m to 8.5-8.8 m. This result is expected. Suppose the distance between a sensor node and an anchor node is 1500 m (this is a normal distance in a $2500 \times 2500 \times 2500$ area), and the speed of sound is 1500 m/s. The sensor node needs at least 2 s to localize itself, in which 1 s for the localization request message traveling 1500 m from the sensor node to the anchor node, and another 1 s for the localization reply message traveling 1500 m from the anchor node to sensor node. within this 2 s, the sensor node may already move 4 m in certain direction unintentionally by current. Therefore when the speed of current is 2 m/s, the average localization error is about 3.2-3.4 m, and when the speed of current is 4 m/s, the average localization error raises to 8.5-8.8 m. In deed, the speed of current influences average localization error a lot in EELM scheme, but this happens in other models as well if these models do not prevent the influences intentionally.

In summary, indeed the environment bring influences in models. But it does not influence the gap of energy consumption between EELM and OLTC. In a sense, this result supports our conclusion that EELM is more energy-efficient than OLTC, especially for sensor nodes.

6.7 Conclusion

In this chapter, we briefly introduce 5 models for comparison, in which EELM-Min and EELM-Max schemes are the easy version of EELM and used as the lower and upper bound of the performance results. To measure the performances we set 6 performance metrics, they describe localization coverage, energy efficiency, localization delay, and localization error. As the results, we can see that in localization coverage, localization delay, and localization error, the performances of EELM and EELM-Fy are close to that of OLTC. However, sensor node energy efficiency of EELM and EELM-Fy are much better than that of OLTC, and the average energy consumption per node of OLTC is higher than that of EELM and EELM-Fy. For energy consumption of anchor nodes, OLTC has more advantage than EELM and EELM-Fy due to the strategy and the purpose of EELM and EELM-Fy is to save more energy for sensor nodes by sacrifice some energy in anchor nodes. Finally, we simulate the different speed of current and see the influences on the performance of ELLM. The result is that the speed of current brings nearly no influence to EELM.

Chapter 7

Conclusion

"The more learn, the more found his ignorance"

Rene Descartes

We propose EELM scheme, in which the Single-Leader-Multi-Follower Stackelberg game is used to model the interaction between sensor and anchor nodes. When the game reaches Stackelberg Equilibrium, sensor and anchor nodes get the optimized transmission power to send "Request" and "Reply" messages. In addition, By introducing the mechanism of the "two-hop" neighbor, the sensor nodes get more advantages in the game. It fits the purpose of saving more energy in sensor nodes, because sensor nodes usually carry limited batteries, and some sensor nodes are disposable. Saving energy is expanding the lifetime of the sensor nodes. We also propose an extended version of EELM, which is EELM-Fy. EELM-Fy replaces the pair of utility functions for sensor and anchor node with a pair of FISs, but the FISs are trained by the data collected from EELM. In the simulation results, we can see that the energy efficiency of EELM and EELM-Fy is higher than that of OLTC respectively.

7.1 Discussion

One of the disadvantages of EELM-Fy is that it costs more resources including time and energy in disposable sensor nodes. Because before using EELM-Fy, we need to collect train data, which costs energy in disposable sensor nodes, and do the training process in Matlab, then install new EELM-Fy scheme to sensor and anchor nodes. Try to automatize all process is also a future work.

One may ask that as the anchor nodes provide service and can charge themselves, why anchor nodes do not send their coordinations with maximum transmission power at the beginning. Because sensor nodes are in sleep mode, anchor nodes need to send wake up message first. Then anchor nodes wait for the "Request" messages from sensor nodes, and only when there is

requirement for localization, anchor nodes reply them. Except the consideration of energy, another reason for not using maximum transmission power to send messages is that the communication channel resource is limited. Signals may collide with higher probability when all nodes use maximum transmission power. Adding frequency division multiplexing techniques can solve this problem.

7.2 Future Work

In some special situation (e.g. storm), the anchor nodes cannot recharge themselves or other energy crises occurs, anchor nodes have limited energy. Anchor nodes can change to OLTC model to save energy for themselves. When anchor nodes have enough energy storage they use EELM model to dedicate themselves to help localization for sensor nodes. There is a simple way to combine OLTC and EELM. When the anchor nodes send first wake up message, the message contains a flag to tell the sensor nodes that which model should be used at this time for localization. And anchor nodes can decide that flag according to their remaining energy to switch the models.

Utility function for anchor nodes (Equation 3.2) only consider the information come from “Request" messages. We also can add the factor which extracted from the neighbor information of the anchor nodes. By doing so, the anchor nodes become co-operative in the game. It is worth to try if this modification brings better performance. However, because the utility function is changed, Stackelberg Equilibrium needs to be verified again.

One future work direction is that when the sensor nodes have enough energy (some sensor nodes can recharge themselves from currents or waves), the sensor nodes can transform to anchor nodes to help other sensor nodes localize themselves. If sensor node can transform to anchor node, the structure of EELM would change a lot.

Bibliography

[1] I. F. Akyildiz, D. Pompili, and T. Melodia, "Underwater acoustic sensor networks: research challenges," *Ad hoc networks*, vol. 3, no. 3, pp. 257–279, 2005.

[2] H.-P. Tan, R. Diamant, W. K. Seah, and M. Waldmeyer, "A survey of techniques and challenges in underwater localization," *Ocean Engineering*, vol. 38, no. 14-15, pp. 1663–1676, 2011.

[3] M. Erol-Kantarci, H. T. Mouftah, and S. Oktug, "A survey of architectures and localization techniques for underwater acoustic sensor networks," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 3, pp. 487–502, 2011.

[4] G. Tuna and V. C. Gungor, "A survey on deployment techniques, localization algorithms, and research challenges for underwater acoustic sensor networks," *International Journal of Communication Systems*, vol. 30, no. 17, pp. 1–21, 2017.

[5] M. Erol-Kantarci, H. T. Mouftah, and S. Oktug, "Localization techniques for underwater acoustic sensor networks," *IEEE Communications Magazine*, vol. 48, no. 12, pp. 152–158, December 2010.

[6] S. Misra, T. Ojha, and A. Mondal, "Game-theoretic topology controlfor opportunistic localization in sparse underwater sensor networks," *IEEE transactions on mobile computing*, vol. 14, no. 5, pp. 990–1003, 2015.

[7] L. Freitag, M. Grund, S. Singh, J. Partan, P. Koski, and K. Ball, "The whoi micro-modem: an acoustic communications and navigation system for multiple platforms," in *Proceedings of OCEANS 2005 MTS/IEEE*, vol. 2, Sept 2005, pp. 1086–1092.

[8] M. T. Isik and O. B. Akan, "A three dimensional localization algorithm for underwater acoustic sensor networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 9, pp. 4457–4463, September 2009.

[9] M. Erol, L. F. M. Vieira, and M. Gerla, "Localization with dive'n'rise (dnr) beacons for underwater acoustic sensor networks," in *Proceedings of the Second Workshop on Underwater Networks*, ser. WuWNet '07. New York, NY, USA: ACM, 2007, pp. 97–100. [Online]. Available: http://doi.acm.org/10.1145/1287812.1287833

[10] M. Erol, L. F. M. Vieira, A. Caruso, F. Paparella, M. Gerla, and S. Oktug, "Multi stage underwater sensor localization using mobile beacons," in *2008 Second International Conference on Sensor Technologies and Applications (sensorcomm 2008)*, Aug 2008, pp. 710–714.

[11] Z. Zhou, J.-H. Cui, and S. Zhou, "Localization for large-scale underwater sensor networks," in *NETWORKING 2007. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 108–119.

[12] M. Erol-Kantarci, S. Oktug, L. Vieira, and M. Gerla, "Performance evaluation of distributed localization techniques for mobile underwater acoustic sensor networks," *Ad Hoc Networks*, vol. 9, no. 1, pp. 61–72, 2011.

[13] X. Cheng, H. Shu, Q. Liang, and D. H. C. Du, "Silent positioning in underwater acoustic sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 3, pp. 1756–1766, May 2008.

[14] X. Cheng, H. S. H. Shu, and Q. Liang, "A range-difference based self-positioning scheme for underwater acoustic sensor networks," in *International Conference on Wireless Algorithms, Systems and Applications (WASA 2007)*, Aug 2007, pp. 38–43.

[15] X. Cheng, A. Thaeler, G. Xue, and D. Chen, "Tps: a time-based positioning scheme for outdoor wireless sensor networks," in *IEEE INFOCOM 2004*, vol. 4, March 2004, pp. 2685–2696.

[16] W. Cheng, A. Thaeler, X. Cheng, F. Liu, X. Lu, and Z. Lu, "Time-synchronization free localization in large scale underwater acoustic sensor networks," in *2009 29th IEEE International Conference on Distributed Computing Systems Workshops*, June 2009, pp. 80–87.

[17] D. Fudenberg and J. Tirole, "Game theory," *Cambridge, Massachusetts*, vol. 393, no. 12, p. 80, 1991.

[18] D. Yang, G. Xue, J. Zhang, A. Richa, and X. Fang, "Coping with a smart jammer in wireless networks: A stackelberg game approach," *IEEE Transactions on Wireless Communications*, vol. 12, no. 8, pp. 4038–4047, 2013.

[19] M. Stojanovic, "Underwater acoustic communications," in *Proceedings of Electro/International 1995*, Jun 1995, pp. 435–440.

[20] J.-H. Cui, J. Kong, M. Gerla, and S. Zhou, "The challenges of building mobile underwater wireless networks for aquatic applications," *IEEE Network*, vol. 20, no. 3, pp. 12–18, May 2006.

[21] J. Heidemann, W. Ye, J. Wills, A. Syed, and Y. Li, "Research challenges and applications for underwater sensor networking," in *IEEE Wireless Communications and Networking Conference, 2006. WCNC 2006.*, vol. 1, April 2006, pp. 228–235.

[22] J. Lloret, S. Sendra, M. Ardid, and J. J. P. C. Rodrigues, "Underwater wireless sensor communications in the 2.4 ghz ism frequency band," *Sensors*, vol. 12, no. 4, pp. 4237–4264, 2012. [Online]. Available: http://www.mdpi.com/1424-8220/12/4/4237

[23] G. Burrowes and J. Y. Khan, "Short-range underwater acoustic communication networks." InTech, 2011, pp. 174–198.

[24] D. B. Kilfoyle and A. B. Baggeroer, "The state of the art in underwater acoustic telemetry," *IEEE Journal of Oceanic Engineering*, vol. 25, no. 1, pp. 4–27, Jan 2000.

[25] J. Partan, J. Kurose, and B. N. Levine, "A survey of practical issues in underwater networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 11, no. 4, pp. 23–33, 2007.

[26] F. Lu, S. Lee, J. Mounzer, and C. Schurgers, "Low-cost medium-range optical underwater modem: Short paper," in *Proceedings of the Fourth ACM International Workshop on UnderWater Networks*, ser. WUWNet '09. New York, NY, USA: ACM, 2009, pp. 1–4. [Online]. Available: http://doi.acm.org/10.1145/1654130.1654141

[27] K. V. Mackenzie, "Nine-term equation for sound speed in the oceans," *The Journal of the Acoustical Society of America*, vol. 70, no. 3, pp. 807–812, 1981.

[28] I. F. Akyildiz, D. Pompili, and T. Melodia, "Challenges for efficient communication in underwater acoustic sensor networks," *SIGBED Rev.*, vol. 1, no. 2, pp. 3–8, Jul. 2004. [Online]. Available: http://doi.acm.org/10.1145/1121776.1121779

[29] M. Stojanovic and J. Preisig, "Underwater acoustic communication channels: Propagation models and statistical characterization," *IEEE Communications Magazine*, vol. 47, no. 1, pp. 84–89, 2009.

[30] M. Stojanovic, "Ofdm for underwater acoustic communications: Adaptive synchronization and sparse channel estimation," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, March 2008, pp. 5288–5291.

[31] S. Milica, "Low complexity ofdm detector for underwater acoustic channels," in *OCEANS 2006*, Sept 2006, pp. 1–6.

[32] G. Mao, B. Fidan, and B. D. Anderson, "Wireless sensor network localization techniques," *Computer Networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

[33] K. Langendoen and N. Reijers, "Distributed localization in wireless sensor networks: a quantitative comparison," *Computer Networks*, vol. 43, no. 4, pp. 499–518, 2003.

[34] S. Čapkun, M. Hamdi, and J.-P. Hubaux, "Gps-free positioning in mobile ad hoc networks," *Cluster Computing*, vol. 5, no. 2, pp. 157–167, 2002.

[35] A. Caruso, S. Chessa, S. De, and A. Urpi, "Gps free coordinate assignment and routing in wireless sensor networks," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 1, March 2005, pp. 150–160.

[36] V. Chandrasekhar and W. Seah, "An area localization scheme for underwater sensor networks," in *OCEANS 2006 - Asia Pacific*, May 2006, pp. 1–8.

[37] D. Mirza and C. Schurgers, "Collaborative localization for fleets of underwater drifters," in *OCEANS 2007*, Sept 2007, pp. 1–6.

[38] H. Luo, Z. Guo, W. Dong, F. Hong, and Y. Zhao, "Ldb: Localization with directional beacons for sparse 3d underwater acoustic sensor networks." *Journal of Networks*, vol. 5, no. 1, pp. 28–38, Jan 2010.

[39] Z. Zhou, Z. Peng, J.-H. Cui, Z. Shi, and A. Bagtzoglou, "Scalable localization with mobility prediction for underwater sensor networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 3, pp. 335–348, 2011.

[40] MathWorks, "Fuzzy inference process," Available: https://cn.mathworks.com/help/fuzzy/fuzzy-inference-process.html, 2018, [Online]. Accessed: Feburary 21, 2018.

[41] tutorialspoint.com, "Functional blocks of fis," Available: https://www.tutorialspoint.com/fuzzy_logic/images/fis_functional_blocks.jpg, 2018, [Online]. Accessed: Feburary 21, 2018.

[42] A. F. Harris, III and M. Zorzi, "Modeling the underwater acoustic channel in ns2," in *Proceedings of the 2Nd International Conference on Performance Evaluation Methodologies and Tools*, ser. ValueTools '07. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 18:1–18:8. [Online]. Available: http://dl.acm.org/citation.cfm?id=1345263.1345286

[43] M. Sugeno, *Industrial Applications of Fuzzy Control*. New York, NY, USA: Elsevier Science Inc., 1985.

[44] J. S. R. Jang, "Anfis: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665–685, May 1993.

[45] A. Kaur and A. Kaur, "Comparison of fuzzy logic and neuro-fuzzy algorithms for air conditioning system," *International Journal of Soft Computing and Engineering*, vol. 2, no. 1, pp. 417–20, 2012.

[46] J. Rada-Vilela, "A fuzzy logic control library in c++," in *Proceedings of the Open Source Developers Conference*, Oct 2013.

[47] nsnam.org, "ns-3 tutorial introduction," Available: https://www.nsnam.org/docs/tutorial/html/introduction.html#about-ns3, 2010, [Online]. Accessed: August 19, 2017.

[48] ——, "ns-3 document src/uan/model/uan-net-device.cc," Available: https://www.nsnam.org/doxygen/uan-net-device_8cc_source.html, 2009, [Online]. Accessed: August 21, 2017.

[49] ——, "ns-3 document src/uan/model/uan-mac-cw.cc," Available: https://www.nsnam.org/doxygen/uan-mac-cw_8cc_source.html, 2009, [Online]. Accessed: August 21, 2017.

[50] ——, "ns-3 document src/uan/model/uan-phy-gen.cc," Available: https://www.nsnam.org/doxygen/uan-phy-gen_8cc_source.html, 2009, [Online]. Accessed: August 21, 2017.

[51] ——, "ns-3 document src/uan/model/uan-transducer-hd.cc," Available: https://www.nsnam.org/doxygen/uan-transducer-hd_8cc_source.html, 2009, [Online]. Accessed: August 21, 2017.

[52] ——, "ns-3 document src/uan/model/uan-channel.cc," Available: https://www.nsnam.org/doxygen/uan-channel_8cc_source.html, 2009, [Online]. Accessed: August 21, 2017.

[53] ——, "ns-3 document src/uan/model/uan-tx-mode.cc," Available: https://www.nsnam.org/doxygen/uan-tx-mode_8cc_source.html, 2009, [Online]. Accessed: August 21, 2017.

[54] R. J. Urick, *Principles of Underwater Sound 3rd Edition*, 3rd ed. Peninsula Publishing, August 2013.

[55] nsnam.org, "ns-3 document src/uan/modeluan-prop-model-thorp.cc," Available: https://www.nsnam.org/doxygen/uan-prop-model-thorp_8cc_source.html, 2009, [Online]. Accessed: August 21, 2017.

[56] M. Stojanovic, "On the relationship between capacity and distance in an underwater acoustic communication channel," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 11, no. 4, pp. 34–43, Oct. 2007. [Online]. Available: http://doi.acm.org/10.1145/1347364.1347373

[57] L. M. Brekhovskikh, "Scattering of sound at rough surfaces," in *Fundamentals of Ocean Acoustics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 182–225. [Online]. Available: https://doi.org/10.1007/978-3-662-07328-5_9

[58] N. Parrish, L. Tracy, S. Roy, P. Arabshahi, and W. L. J. Fox, "System design considerations for undersea networks: Link and multiple access protocols," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 9, pp. 1720–1730, December 2008.

[59] R. M. Buehrer, "Code division multiple access (cdma)," *Synthesis Lectures on Communications*, vol. 1, no. 1, pp. 1–192, 2006.

[60] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Transactions on Networking (ToN)*, vol. 12, no. 3, pp. 493–506, 2004.

[61] H. Karl and A. Willig, *Protocols and architectures for wireless sensor networks*. John Wiley & Sons, 2007.

[62] S. F. Alam, Y. Rafat, and H. Bilgrami, "Acoustical analysis of aligarh muslim university's central auditorium," *Proceedings of Meetings on Acoustics*, vol. 28, no. 1, pp. 15–21, 2016. [Online]. Available: https://asa.scitation.org/doi/abs/10.1121/2.0000438