Analysis of Algorithms for Multi-Modal Communications in Underwater Sensor Networks

Elizabeth Basha[†], Nicholas Yuen[†], Michael O'Rourke[†], and Carrick Detweiler[*]
[†]University of the Pacific, Stockton, California
[*]University of Nebraska–Lincoln, Lincoln, Nebraska
ebasha@pacific.edu,n_yuen@u.pacific.edu,m_orourke@u.pacific.edu,carrick@cse.unl.edu

ABSTRACT

Acoustic communication typically dominates the power usage in underwater sensor networks. To balance the conflicting needs of communication and power usage, we utilize a sensor network platform with underwater acoustic communication, surface level radio communication, and a depth adjustment system to switch between them. We focus on determining which nodes should surface to create a radio communication route for situations where, for the message size, it is more energy efficient to transmit via radio yet acoustic messaging and the depth adjustment system still dominate the overall energy usage. For a given path in 2D or 3D, we develop and examine a set of eight algorithms to select these nodes while taking energy usage and packet loss into account. We perform an analysis of the algorithms and show that a decentralized approach where nodes know their two-hop neighbors provides the most energy efficient method.

1. INTRODUCTION

Underwater sensor networks struggle to balance the need for communication with the limitations of fixed energy systems. Without the ability to recharge their batteries, underwater nodes decrease their lifetime with every action: sensing, computation, and communication. Of these operations, communication uses the most energy yet is necessary for any system-wide actions including transmitting the data to off-shore sites.

Typically acoustic communication provides the transmission method underwater; however, its slow data rate and high energy limit message size and quantity. Combining acoustic with other methods (surface nodes with radio or satellite, data muling AUVs, and others) allows for balancing the energy and communication needs. This still has some limitations as it requires differing types of nodes - those underwater using acoustics and different nodes (or robots) using alternative communication methods.

However, it is possible to use nodes with two forms of communication: (1) slow data rate, underwater acoustics and (2) higher data rate, surface radio. Our AquaNode system utilizes both communication methods [7]. The AquaNodes communication via the radio through surfacing using a depth adjustment system. This still provides a challenge as surfacing has a high initial energy cost. To balance the initial energy output with the significant increase in data per Joule when using the radio, the system needs to optimize the number of nodes surfacing during message communication. The system also needs to ensure that this number of nodes creates a connected path between the nodes that want to communicate. Therefore, we need algorithms that achieve this minimization of energy while ensuring connectivity.

This work builds on our prior work that developed equations to determine trade-offs between the different energy uses [10]. In that work, we determined that, for typical depths of operation, once a message exceeds 1328bits, it is more energy efficient to transmit the message via the surface radio system instead of acoustically. We began to explore how to create a surface radio route to transmit the message, developing four different algorithms and performing a preliminary analysis based on time and movement.

In this paper, we develop a set of eight algorithms to explore different approaches from fully decentralized to centralized. We focus on the situation where our message is larger than 1328bits, but smaller than approximately 1920Mbits. Within this range, finding the most energy efficient route depends on the acoustic communication and depth adjustment systems as they dominate the energy usage. Once the data transmission exceeds 1920Mbits, however, the radio communication begins to dominate the energy usage, leaving an acoustically expensive approach (e.g. Dijkstra's shortest path) as the most efficient.

Our eight algorithms attempt to balance acoustic communication with depth adjustment to find the best approach. We begin by exploring the two-dimensional scenario where the nodes exist in the same line on the y-axis. We then expanded this scenario to include packet loss. This impacts the algorithms differently as some require more communication than others. We examine the algorithms with no packet loss, 44% loss (typical for the AquaNodes), and 80% loss. We also explore the algorithms when extended to full three-dimensional scenarios.

WUWNet '14, November 12-14, 2014 Rome, Italy
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3277-4/14/11:.$15.00
http://dx.doi.org/10.1145/2578726.2578744

Overall, we examine both centralized and decentralized algorithms. The centralized algorithms are used to establish performance baselines, but are less useful in practice as they require global knowledge. All scenarios provide approximately the same algorithmic ordering. Of the decentralized algorithms, if we want to minimize acoustic communication and node movement, our best approach is *Look-Ahead*, a decentralized approach relying on prior knowledge of neighbors and the neighbors' neighbors. In the absence of this two-hop information, we can use *Furthest Radio*, which is also decentralized, but only requires knowledge of one-hop neighbors.

We organize the paper as follows. Section 2 outlines related work. Section 3 describes the hardware system we use to experimentally determine real world parameters and Section 4 describes our algorithms and simulation environment that allows us to explore a range of scenarios. Section 5 explains the test results of the algorithms in two-dimensional topologies and analyzes the algorithm performance. Section 6 explores the impact of acoustic packet loss. Section 7 then expands the simulator to deploy the nodes in three-dimensional topologies and determines the impact of this change. Finally, Section 8 summarizes the results and future work.

2. RELATED WORK

In this section we discuss related work in surface nodes as part of underwater systems and route planning. Many underwater sensor networks leverage surface nodes for long-range, high-throughput communication channels [2]. One example is the US Navy's SeaWeb system that had a number of radio/acoustic ("Racom") nodes at the surface that could communicate both acoustically and with radio to satellites, ships, or shore [11]. For systems with underwater nodes, using a surface gateway node is one of the more practical methods to obtain information from an underwater network [5]. Placement of the gateway nodes in order to minimize energy and end-to-end delay for a given set of underwater nodes has been examined using, for example, integer linear programming [15]. With these types of systems, the overall bandwidth of the system is limited by the acoustic channel, since all nodes need to transmit acoustically to a radio gateway node. Another option for obtaining data from an underwater sensor network is to use an underwater vehicle to collect the data and transmit it back when at the surface. This has been demonstrated with energy efficient underwater gliders that surface periodically to send back data and obtain position fixes [3].

Our system differs from these in that our underwater nodes can choose to surface to send large sets of data themselves or to act as a relay for other nodes. This eliminates the acoustic channel bottleneck, but requires significant energy to surface and surfacing also removes the node from its desired depth for sensing.

In addition to looking at platforms, we also examine prior work on routing. Underwater, there has been a significant amount of work focused on acoustic route creation [4, 14]. On land, there is also significant work on creating energy efficient routes within sensor networks. Stojmenovic *et al.* examined a wide variety of routing protocols and found that greedy algorithms have performances that rival an optimal shortest path algorithm for dense graphs, but low delivery rates for sparse graphs [12] . Additionally, Stojmenovic *et al.* determined that algorithms with delivery guarantees may have high communication overhead for sparse graphs. Tan *et al.* studied the problem of shortest-path geographic routing in static sensor networks and developed an algorithm based on the construction of a reduced visibility graph to find near optimal paths [13]. We utilize their results to develop similar approaches in our underwater scenario.

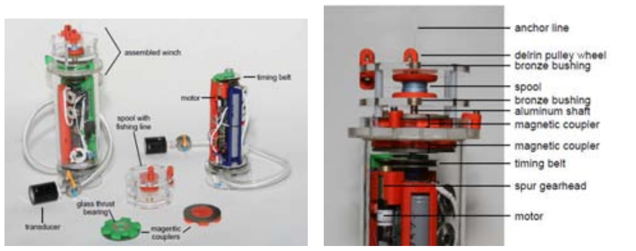

Figure 1: (a) AquaNode (left) and (b) Depth Adjustment System Details (right) [6]

3. AQUANODE PLATFORM

In our prior work we developed the AquaNode underwater sensor network [6, 7]. One of the key features is that the AquaNodes can adjust their depth in the water, communicate using three different methods (acoustic, radio, and optical), and utilize multiple sensors. In normal operation, an AquaNode anchors to the seafloor and floats in the water mid-column in order to sense the environment. However, in order to measure the entire water column or surface for radio communication, AquaNodes have a depth adjustment system that allows the node to dynamically rise and descend in the water column.

Figure 1(a) shows the full AquaNode and the winch based depth adjustment system while Figure 1(b) shows the details of the depth adjustment system. Theoretically, the AquaNodes are capable of radio communication at 57kbit/s within 3km, acoustic communication at 300b/s within 400m, and optical communication at 3Mb/s within 3m [6]. In practice we obtain significantly smaller ranges. We performed basic radio tests between two nodes and found that the first packet losses started after 150m, more major losses occurred after 200m with 50% loss after 300m, and a maximum range of slightly over 500m with high packet loss. In this paper, we assume a 200m range, but, for simplicity of analysis, while we consider packet loss for acoustic communication, we do not for the radio channel as it has no impact on the algorithms.

Acoustic communication is more difficult to characterize and is highly dependent on positioning and channel characteristics. Figure 2 shows the acoustic communication success rate for a pair of AquaNodes placed 36m apart in 10m deep water [7]. The depths of the AquaNodes were changed between 2.0m to 7.0m for Node 0 and 2.0m to 6.0m for Node 1. As seen in the figure, changing the depth can cause packet success rates ranging from under 50% to 95%. In-

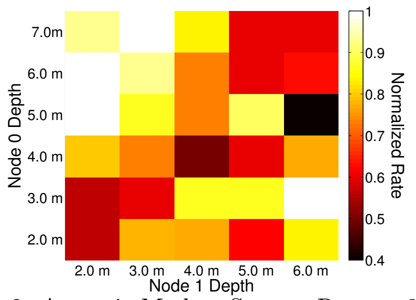

Figure 2: Acoustic Modem Success Rate at 36m [7]

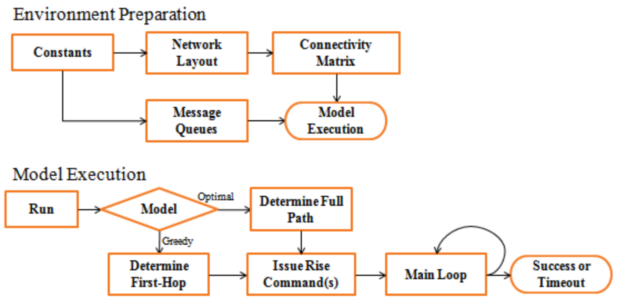

Figure 3: Flowchart of Simulator

terestingly, when both nodes were mid-column, a common deployment strategy, the success rate was at the low end of the spectrum. In typical environments we have found our acoustic modems can work at 100m range with packet-loss of 44% [8], but this is highly dependent on the configuration, so we explore the impact of acoustic packet loss on our algorithms in this paper.

We also measured energy use to define constants used to calculate the energy usage of the different algorithms, depending on how many bits are sent through radio/acoustic communication and how many meters it travels using its depth adjustment system. Table 1 defines the constants [10].

Variable	**Value**
$RADIO\_TX\_ENERGY$	$0.00016 J/bit$
$RADIO\_RX\_ENERGY$	$0.0000457 J/bit$
$ACOUSTIC\_TX\_ENERGY$	$0.1136 J/bit$
$ACOUSTIC\_RX\_ENERGY$	$0.063 J/bit$
$WINCH\_ENERGY$	$15 J/m$

Table 1: AquaNode Power Constants

4. ALGORITHMS

In this section, we describe the simulator and the various communication algorithms we developed.

4.1 Simulator Overview

The simulation environment models multi-hop communication in an underwater sensor network that utilizes AquaNodes. The simulation environment routes a packet of data between two AquaNodes that are on opposite sides of the network. It uses underwater acoustic communication to calculate a multi-hop path between the nodes, the nodes rise, and then use radio communication to send the packet of data across the network. While calculating the path of the packet, the simulation environment tracks a number of metrics such as energy consumption, messages sent, and run-time of the entire process.

We implemented the simulator as a MATLAB procedural environment consisting of a set of scripts and functions [9]. Figure 3 shows the general flow of the simulation environment. There are two main phases of the simulation environment: preparation and execution. In the preparation stage, the simulator defines and initializes constants such as the number of nodes, communication power, and communication range. Then it creates node positions based on a linear topology that follows rules set by the defined constants. The next step creates two connectivity matrices that describe nodes' communication connectivity for radio communication and acoustic communication. These connections are based on AquaNode communication range constants and simulator-created node positions. The communication range constants are based on the communication analysis in Section 3; we use a radio range that is greater than the acoustic range, although this is a configurable parameter. The simulator ensures that each node has at least one acoustic neighbor, which also implies that they are in radio range (since the radio range is larger than the acoustic). The final step of preparation is initializing the message queues.

In the execution stage, the simulator first determines which communication routing algorithm the user specified. If it is a centralized algorithm, the simulator determines the full multi-hop communication path, has the nodes rise, and sends the radio message across the surfaced nodes. If it is a decentralized algorithm, the simulator calculates the multi-hop communication path node by node. The starting node determines the next node in the sequence, communicates its selection to that node, and then rises. Each selected node continues these operations until the destination node is reached. Finally, the simulator forwards the message via radio communication.

4.2 Algorithms

The simulator has eight different communication routing algorithms: *Greedy Furthest Acoustic*, *Greedy Furthest Radio*, *Min-Hop Furthest*, *Greedy Shallowest Acoustic*, *Greedy Shallowest Radio*, *Min-Hop Shallowest*, *Greedy Look-Ahead*, and *Greedy Look-Back*.

Our algorithms are divided into two different categories: acoustic-centric and radio-centric. Algorithms are considered acoustic-centric if they make decisions based on neighboring nodes within acoustic communication range and radio-centric if they make decisions based on neighboring nodes within radio communication range. There are two acoustic-centric algorithms: *Greedy Furthest Acoustic* and *Greedy Shallowest Acoustic*. *Greedy Furthest Acoustic* looks for the acoustic neighbor furthest away from the transmitting node and closest to the destination to minimize the number of nodes surfacing; *Greedy Shallowest Acoustic* looks for the acoustic neighbor closest to the surface to minimize energy costs of surfacing.

We will use *Greedy Furthest Acoustic* to describe the general behavior and basic analysis of each algorithm. This

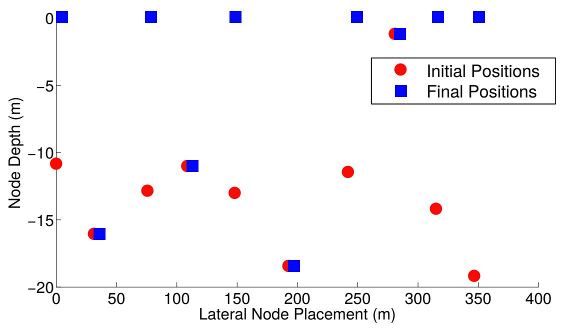

Figure 4: Graph of Start and End Positions for Greedy Furthest Acoustic

algorithm operates using a decentralized approach and a message we call a rise command. Rise commands are acoustic messages that contain no data; the fields of the packet are destination, source, and a protocol/command field corresponding to "rise." Upon receiving a rise command, a node will compute the next node to rise and send it a rise command. This process repeats until the destination node receives a rise command and completes the path for the radio message. Algorithm 1 outlines the high-level process and the computation of the next node for *Greedy Furthest Acoustic*. All greedy algorithms behave the same way; the only change between algorithms is how next-neighbor's are selected.

Algorithm 1 Overview of Greedy Furthest Acoustic

```
loop
    receive rise command $P$
    if $P$.Destination $\equiv$ Self then
        exit
    else
        queue P for delayed-transmission
        find furthest acoustic neighbor
        send rise command
    end if
    rise
end loop
```

Figure 4 shows a sample topology and the path *Greedy Furthest Acoustic* created. If each node has only one forward-neighbor available, one neighbor closer to the destination, then this algorithm would require all nodes to rise. Nodes being spaced evenly near the edge of acoustic range would create a worst-case topology for this algorithm, as this is the definition of having only one forward-neighbor. Nodes placed close together or acoustic range long enough to allow for multiple forward-neighbors allow this algorithm to see significant improvement in performance.

	Number of Nodes			
	25	**50**	**75**	**100**
Avg. Dist (m)	6.5131	6.7744	6.7548	6.6132
Avg. Depth (m)	-10.1220	-9.9952	-9.9465	-10.0011
Avg. Energy (J)	90.53	89.50	88.57	88.26
Avg. Time (μs)	62.2138	62.6475	61.9471	61.8014

Table 2: Sample Statistics for Greedy Furthest Acoustic

Table 2 summarizes the performance of the model implementing this algorithm, averaged over 500 runs. The Distance field represents the total distance traveled by nodes in the network averaged across all nodes including those that do not participate in routing. Depth is the average starting depth for all nodes. Energy consumption is averaged for all nodes in a network, with most energy being consumed by node movement. Average time represents the amount of time it took for each node to complete a single iteration through the simulation.

Immediately clear is that the average depth is always around -10 meters; we expect this as the range is from -20m to 0m depth and, after sufficient randomization, we should observe an average depth of -10m. Somewhat surprising to note is the relatively small change in average distance and energy. This implies that the load on all nodes in the network is agnostic to network size. Additionally, the energy and time metrics scale well. We perform a similar analysis on *Greedy Shallowest Acoustic* and see similar results (details in [9]).

The other six algorithms are radio-centric and consist of: *Greedy Furthest Radio*, *Greedy Shallowest Radio*, *Greedy Look-Ahead*, *Greedy Look-Back*, *Min-Hop Furthest*, and *Min-Hop Shallowest*. *Greedy Furthest Radio* and *Greedy Shallowest Radio* perform similar to their acoustic counterparts, but using radio neighbors instead of acoustic.

Min-Hop Furthest and *Min-Hop Shallowest* are the centralized approaches where the starting node computes everything. Both use Dijkstra's to determine the set of nodes that rise with *Min-Hop Furthest* being the unweighted version and *Min-Hop Shallowest* being the weighted version that uses depth for the weights.

Since the centralized approach has a significant acoustic communication cost, we also develop *Greedy Look-Ahead*, which uses connection information spanning out to the furthest radio neighbor's furthest neighbor. With this information, it calculates the optimal next step using Dijkstra's algorithm with link-costs being weighted by node depth. The algorithm selects the minimum weight path, which serves as an approximation of the globally optimal route. The only requirements placed on the system are: no-backwards traversals (messages always advance) and the path chosen must have a length greater than two (not including the sender). The path length requirement provides a safe-guard against becoming a purely shallowest-neighbor approach.

Greedy Look-Back is a time saving version of *Greedy Look-Ahead*. This algorithm sends a rise command to the furthest radio neighbor. Upon receipt of a rise command, the receiving node will check to see if it was the shallowest neighbor of the sending node. If so, it will rise and the algorithm continues forward. If not, it will invoke the look-ahead algorithm on behalf of the original sender, determine the appropriate node, and send a "forced-rise" command. The forced-rise can not be further regressed; the receiving node is already part of the locally optimal path.

5. 2D TOPOLOGY EXPERIMENTS

We first explore how the algorithms perform when the nodes exist along the same y-axis line but at different depths. The experiments measure and compare the energy efficiency of the algorithms for a default set of settings.

System Default Values	
Acoustic Range (m)	100
Radio Range (m)	200
Minimum X-axis Spacing (m)	30
Maximum X-axis Spacing (m)	60
Minimum Y-axis Spacing (m)	0
Maximum Y-axis Spacing (m)	0
Maximum Depth (m)	20

Table 3: Default Values for System Variables for 2D Experiments

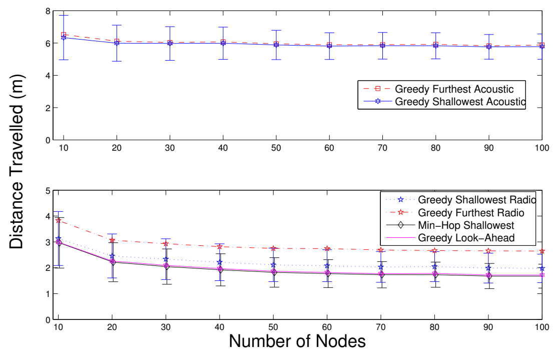

Figure 5: Graph of Start and End Positions

The simulator requires several variables to be defined beyond the number of nodes and length of message queues. These variables include minimum and maximum lateral node placement (x-axis), minimum and maximum node depth (z-axis), and the range of the acoustic and radio modems. Table 3 describes the default values used in the experiment for these variables. These values ensure that each node has at least one acoustic neighbor closer to the destination than itself and, depending on a node's placement in the network, at least three radio neighbors closer to the destination than itself.

We ran the eight algorithms on network sizes from ten to 100 nodes, in increments of ten, across 500 unique topologies. Figure 5 shows the average amount of motion any node could expect to move for a certain network size. The error bars on the plots represent standard deviation and are only shown for algorithms that are significantly different from each other. We choose to separate acoustic and radio algorithms because the acoustic algorithms require more movement, placing them on a different scale. The upper plot contains data on the acoustic-centric algorithms with *Greedy Shallowest Acoustic* showing estimates on error. We see minimal difference between the two acoustic algorithms. The lower plot contains results on all but two radio-centric algorithms (excluded are *Greedy Look-Back* and *Min-Hop Furthest* as they are similar to *Greedy Look-Ahead* and *Min-Hop Shallowest* respectively). In this plot, we see that *Min-Hop Shallowest* and *Greedy Look-Ahead* require the least movement, indicating a trade-off between computational complexity and average node movement.

We might expect the energy comparison to mirror these results as we know that the depth adjustment system uses a large amount of energy. Figure 6 shows the average to-

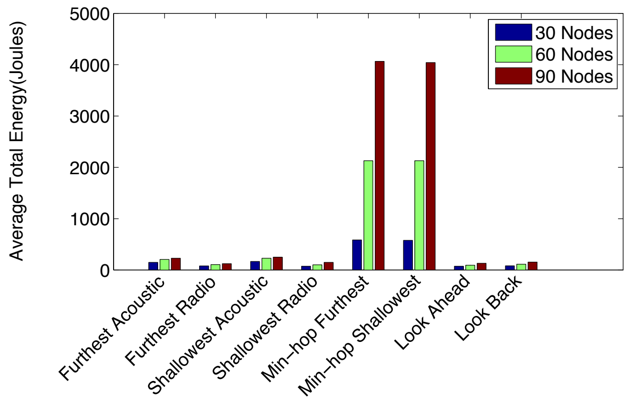

Figure 6: Average Total Energy for All Algorithms

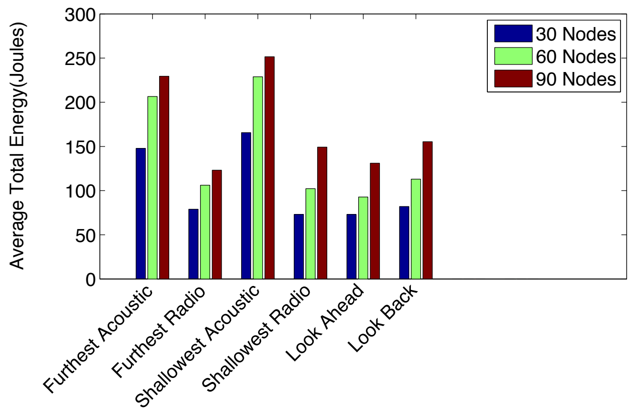

Figure 7: Average Total Energy for Decentralized Algorithms Only

tal energy for all the algorithms, showing a different result. Here we see the two centralized approaches (*Min-Hop Furthest* and *Min-Hop Shallowest*) require significantly more energy than any of the decentralized approaches. This is due to the two centralized algorithms requiring much more communication than the decentralized algorithms because, for the centralized approaches, every message generated is sourced from the first node. To examine the decentralized algorithms more closely, we remove the two centralized approaches and see the result in Figure 7. The two acoustic approaches, which both require more movement than the other four, also require more energy. Of the remaining radio algorithms, *Look Ahead* performs the best for 30 and 60 nodes, but, once the number of nodes increases to 90, performs marginally worse than *Furthest Radio*.

6. ACOUSTIC PACKET-LOSS

We now extend the simulation environment to include acoustic packet-loss in order to provide more realistic results. On the AquaNode platform, our experimental results showed 44% packet-loss at typical communication range [8]. In this section, we discuss our implementation of acoustic packet-loss and then analyze its impact on the algorithms.

6.1 Implementation

We implement acoustic packet-loss by including a probability of the message not being received. If the message is not received, the transmitting node resends until the message is correctly received. This approach assumes that the transmission will eventually succeed; to avoid infinite loops in cases where this assumption does not hold, the simulator will time out and fail. The rate at which acoustic packets are successfully transmitted is controlled by a variable ACOUSTIC_SUCCESS. The user sets this variable at the start of every simulation to a value ranging from 0-1 where 0 is a 0% success rate and 1 a 100% success rate.

The simulation updates acoustic transmit energy and acoustic message propagation time after each sent message. Equation 1 shows the acoustic transmit energy update equation where $ACOUSTIC\_TX\_ENERGY$ is the energy used per bit, *pkt_len* is the length of the data packet being transferred in bytes, and messaging overhead is accounted for by the addition by 24 bits.

$$E_{ACS} = E_{ACS} + ACOUSTIC\_TX\_ENERGY * (24 + 8 * pkt\_len) \quad (1)$$

The constant $ACOUSTIC\_TX\_ENERGY$ is set to 0.1136 joules based on experimental AquaNode results [8].

The simulator updates the acoustic message propagation time based on the distance between the communicating nodes multiplied by a constant TX_DELAY; this can be seen in Equation 2. The TX_DELAY value of 0.00067 s/m comes from [1].

$$T_{AMP} = T_{AMP} + N_{dist} * TX\_DELAY \quad (2)$$

Currently, the acoustic message propagation time does not account for acknowledgment time. We make this assumption as acoustic message propagation time is a tool for verifying correct operation and gaining an understanding of the time needed for the network to calculate a communication path. We also added metrics to measure the total number of acoustic messages sent and the total number of failed acoustic messages.

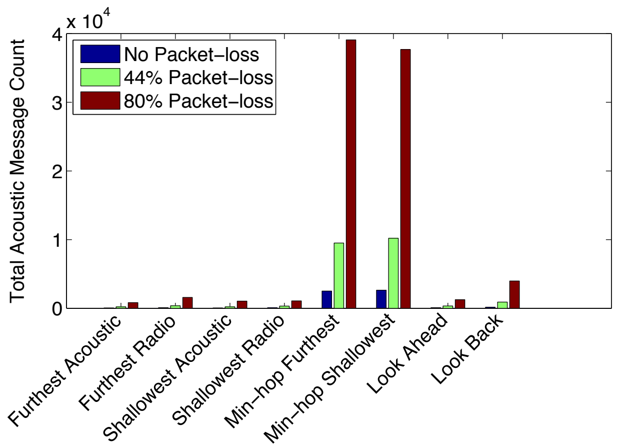

Figure 8: Number of Total Acoustic Messages Sent with Different Acoustic Packet-loss Rates

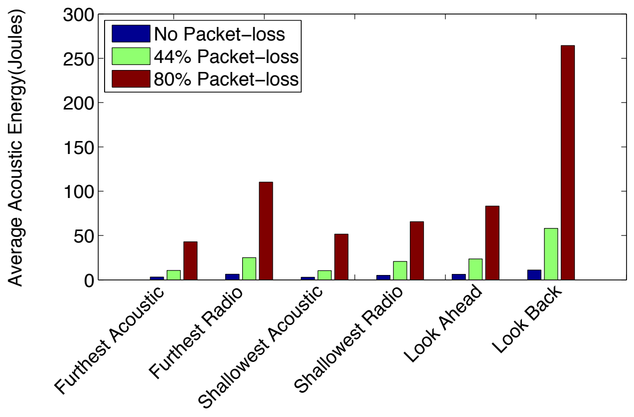

Figure 9: Average Acoustic Energy with Different Acoustic Packet-loss Rates

6.2 Analysis

We analyze the modified simulation environment to ensure it operates as expected and to see if packet loss has any effect on the ordering of the communication algorithms. In our experiment, we use a network size of 60 nodes, skew acoustic packet-loss rates at values of {0%, 44%, 80%}, and average the results of 10 runs. We chose the acoustic packet-loss rates of 0%, 44%, and 80% to gain an understanding of the effect of the variable in the best case scenario (0%), experimentally measured scenario (44%), and near worst case scenario (80%). Using 100% acoustic packet-loss renders acoustic communication completely useless and, in turn, breaks the simulation environment. After running the experiment, we analyzed three different parameters: (1) the acoustic message count of the eight different communication algorithms, (2) the acoustic energy consumption of the six decentralized communication algorithms, and (3) the total energy consumption of the six decentralized communication algorithms.

Acoustic message count with packet-loss We first verify that the number of total acoustic packets sent increases as the acoustic packet-loss rate increases.

Figure 8 shows the results, which see the same trend across all algorithms. As the acoustic packet-loss rate increases, the number of total acoustic packets sent increases and scales to expected values. For example, the algorithm *Greedy Shallowest Radio* transmits 77 packets with 0% loss, 326 packets with 44% loss, and 1079 packets with 80% loss. To understand these results, we consider that, on average, each node has three neighboring nodes within acoustic range and must successfully communicate with all of them. This leads us to expect that the number of packets sent for 0% packet-loss will be approximately 18.67% of the number of packets sent for 44% packet-loss; likewise, the number of packets sent for 0% will be 6.67% of the packets sent for 80% packet-loss. The results show that the amount of packets sent for 0% packet-loss are 23.6% of the number of packets sent for 44% packet-loss and 7.1% for 80% packet-loss. This experiment confirms our implementation of packet loss and suggest some trends on network impact, which we now explore.

Acoustic energy consumption with packet-loss The second analysis compares acoustic packet-loss rates and the average amount of acoustic energy used per node in the network for six of the eight algorithms. We chose to remove the centralized algorithms because their energy results were so much larger than the decentralized algorithms that we

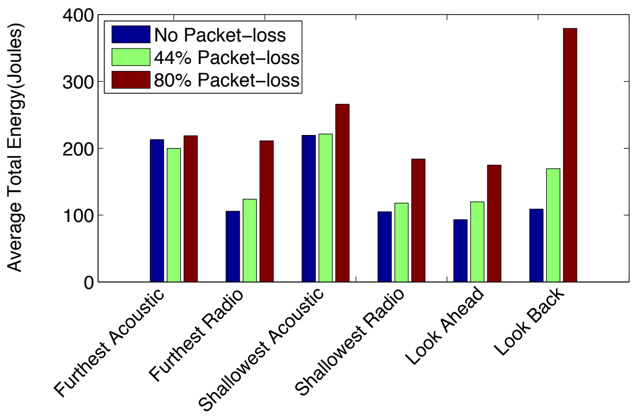

Figure 10: Average Total Energy with Different Acoustic Packet-loss Rates

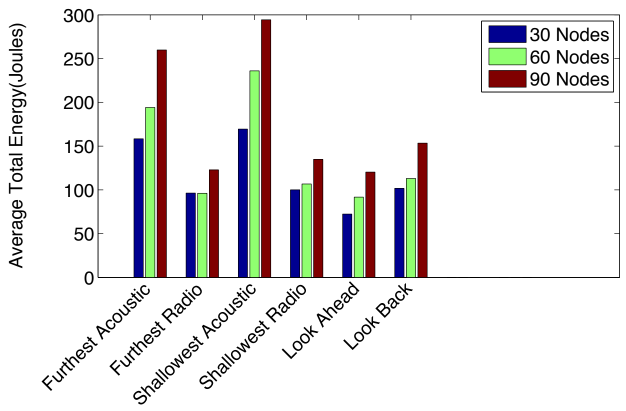

Figure 12: Total Average Energy for Decentralized Communication Algorithms in 3D Topologies

see no reason to implement those on a network.

Figure 9 outlines the results. For all algorithms, average acoustic energy used per node increases as the acoustic packet-loss rate increases. We expect this since the average acoustic energy used per node depends directly on the amount of packets being communicated. In terms of acoustic energy efficiency, the two acoustic-centric communication algorithms are the most efficient as one would expect since those algorithms intentionally maximize that metric.

Total energy consumption with packet-loss Our next analysis explores the energy efficiency of the AquaNode as a whole, considering all energy metrics. The total energy equation consists of processing energy, radio receive energy, acoustic receive energy, radio transmit energy, acoustic transmit energy, and movement energy.

The resulting order of the algorithms in this analysis, seen in Figure 10, differs from the previous result seen in Figure 7. Since this considers all energy metrics, these results reflect the fact that acoustic transmit energy is only a small part of the total energy consumption. Once we consider all energy sinks, the most energy efficient algorithm is a tie between *Shallowest Radio* and *Look Ahead*. These results confirm that acoustic packet-loss has a larger impact on the algorithms that use more acoustic communication.

7. 3D NETWORK TOPOLOGIES

In this section, we discuss the expansion of the simulator and algorithms to support 3D network topologies. Until now, the simulator created only linear network topologies with two dimensions: x (the node's Euclidean x distance away from the start node) and z (node depth). As not all underwater networks are in this configuration, we added the y dimension to explore three-dimensional network topologies.

7.1 Implementation

To add 3D network topologies to the simulation environment, we needed to modify how it generates the topologies and the connectivity matrices. The initial simulator placed nodes in a linear fashion along the xz-plane based on a preexisting topology or in a pseudo-random fashion according to user specified parameters.

To extend the simulation environment to include 3D, we add the y-axis and allow the user to specify the minimum and maximum Euclidean y distance for a node. The simulator then randomly generates nodes with y coordinates within those ranges (along with x and z as originally generated). Connectivity matrices now account for that y-dimension to ensure nodes can reach each other; this is simply an extension of the 2D equation.

7.2 Analysis

We now verify correct operation of 3D topologies and analyze their impact on the ordering of the communication algorithms.

To verify that the simulator correctly generates 3D topologies, we examine the node positions generated. Figure 11(a) shows a typical 2D topology and Figure 11(b) shows a typical 3D topology. In the 2D topology, we see that the node positions are all in a plane where the y position is 0. In the 3D topology, we see that the node positions vary between 30 and 60 meters along the y-axis, correctly generating a 3D topology.

We now analyze the communication algorithms for any changes in their ordering. In these tests, we use network sizes of 30, 60, and 90 nodes and average the results from 10 runs. We ignore the centralized communication algorithms since we determined they were ineffective in Section 4.

Figure 12 shows the total average energy consumption for all decentralized communication algorithms in 3D topologies. The results are nearly identical to the 2D results in Figure 7 with small differences due to the randomization of the network topologies and averaging. Therefore, independent of 2D or 3D topology, our results show that *Look Ahead* provides the most energy efficient algorithm for the computational requirements. It only requires that nodes know the full network topology in order to compute, in a decentralized approach, an energy efficient set of nodes to surface for high data rate communication.

8. CONCLUSION

This paper addresses the problem of determining which subset of nodes should surface in underwater sensor networks with the multiple communication methods and the

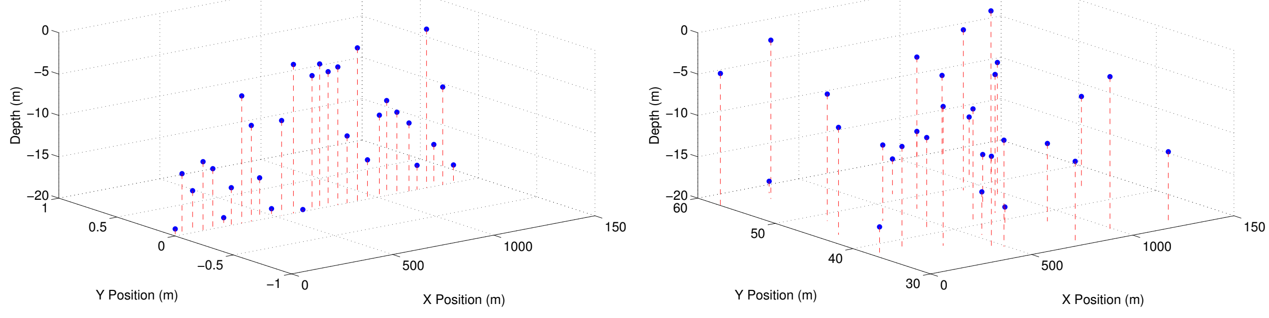

Figure 11: (a) 2D and (b) 3D Network Topologies

ability to adjust depth. We describe eight different algorithms that determine which subset surfaces in order to create a communication path for radio messages. We explore both two-dimensional and three-dimensional topologies with packet loss. All experiments result in the decentralized *Look-Ahead* algorithm providing the most energy efficient approach. If the system does not have the two-hop neighbor information that this approach requires, *Furthest Radio* provides a reasonable alternative.

While our simulation utilized experimental parameters, in the future, we would like to implement the algorithm on the AquaNode network. We also plan to expand the work to support data muling options (both AUV and UAV) and decide on the best approaches given four different options.

9. ACKNOWLEDGMENTS

We are grateful to NSF RI (IIS-1116221) and NSF CNS (CSR-1217400 and CSR-1217428), which partially supported this work.

10. REFERENCES

[1] I. F. Akyildiz, D. Pompili, and T. Melodia. Challenges for efficient communication in underwater acoustic sensor networks. *ACM Sigbed Review*, 1(2):3–8, 2004.

[2] I. F. Akyildiz, D. Pompili, and T. Melodia. State-of-the-art in protocol research for underwater acoustic sensor networks. In *Proceedings of the 1st ACM international workshop on Underwater networks*, WUWNet '06, pages 7–16, New York, NY, USA, 2006.

[3] B. Chen, P. C. Hickey, and D. Pompili. Trajectory-aware communication solution for underwater gliders using WHOI micro-modems. In *IEEE Communications Society Conference on Sensor Mesh and Ad Hoc Communications and Networks (SECON)*, pages 1–9, 2010.

[4] R. Coutinho, L. Vieira, and A. Loureiro. DCR: Depth-controlled routing protocol for underwater sensor networks. In *Computers and Communications (ISCC), 2013 IEEE Symposium on*, pages 4530–458, July 2013.

[5] J. Cui, J. Kong, M. Gerla, and S. Zhou. The challenges of building mobile underwater wireless networks for aquatic applications. *Network, IEEE*, 20(3):12–18, June 2006.

[6] C. Detweiler, M. Doniec, I. Vasilescu, E. Basha, and D. Rus. Autonomous depth adjustment for underwater sensor networks. In *Proc. of the Intl Workshop on UnderWater Networks (WUWNet)*, pages 12:1–12:4, 2010.

[7] C. Detweiler, M. Doniec, I. Vasilescu, and D. Rus. Autonomous depth adjustment for underwater sensor networks: Design and applications. *IEEE Transactions on Mechatronics*, 17(1):16–24, 2012.

[8] C. J. Detweiler. *Decentralized sensor placement and mobile localization on an underwater sensor network with depth adjustment capabilities.* PhD thesis, Massachusetts Institute of Technology, 2010.

[9] M. O'Rourke. Simulating underwater sensor networks and routing algorithms in matlab. Master's thesis, University of the Pacific, 2012.

[10] M. O'Rourke, E. Basha, and C. Detweiler. Short paper: Multi-modal communications in underwater sensor networks using depth adjustment. In *Proceedings of the Seventh ACM International Conference on Underwater Networks and Systems*. ACM, 2012.

[11] J. Rice. SeaWeb acoustic communication and navigation networks. In *Proceedings of the International Conference on Underwater Acoustic Measurements: Technologies and Results*, 2005.

[12] I. Stojmenovic. Position-based routing in ad hoc networks. *IEEE Communications Magazine*, 40(7):128–134, 2002.

[13] G. Tan, M. Bertier, and A. Kermarrec. Visibility-Graph-Based Shortest-Path geographic routing in sensor networks. In *IEEE INFOCOM*, pages 1719–1727, Apr. 2009.

[14] H. Yan, Z. J. Shi, and J.-H. Cui. Dbr: Depth-based routing for underwater sensor networks. In *Proceedings of the 7th International IFIP-TC6 Networking Conference on AdHoc and Sensor Networks, Wireless Networks, Next Generation Internet*, NETWORKING'08, pages 72–86, Berlin, Heidelberg, 2008. Springer-Verlag.

[15] Z. Zhou, H. Yan, S. Ibrahim, J. Cui, Z. Shi, and R. Ammar. Enhancing underwater acoustic sensor networks using surface radios: Issues, challenges and solutions. In G. Ferrari, editor, *Sensor Networks*, Signals and Communication Technology, pages 283–307. Springer Berlin Heidelberg, 2009.