Comparative Analysis of Routing Algorithm for Simulated underwater sensor network

ISSN (e) 2520-7393
ISSN (p) 2521-5027
Received on 27th Nov, 2018
Revised on 26th Dec, 2018
www.estirj.com

Shaharyar Ahmed Abro[1], Tariq Jamil Saifullah Khanzada[2], Shahbaz Wahab [3], Muhammad Mansoor[4]

[1]*Department of Computer Information Engineering Mehran University Engineering &Technology, Jamshoro*
[2]*Department of Computer Systems Engineering Mehran University Engineering &Technology, Jamshoro*
[3,4]*Department of Information Technology, Mehran University Engineering &Technology, Jamshoro.*

Abstract: The new research domain researchers are working on is underwater networks and this topic is taking boost day by day in the research community. Land based approaches of network structure from detection to cutting-edge routing are all well recognized. It is due to the inimitable limitations of operating in an environment of underwater, several of these tried methods need alterations if they function at all. Peer detection and low-level networking have been apportioned by earlier research. In this paper we described and evaluated eight conceivable routing systems with diverse system-acquaintance necessities. It is shown that with a slight set of statistics it is conceivable to attain near-optimal consequences with energy costs significantly lower than centralized optimal algorithms. Further all the three techniques are compared with each other so that a better algorithm can be observed in time of practical implementation.

Keywords: *Wireless Underwater Networks, Acoustic-Centric Algorithms, Radio-Centric Algorithms, Precoding, Simulation*

1. Introduction

Over 70% of the planet carries water and very slight of that is discovered, further we probably recognize even lesser about underwater. A very efficient way to explore the world of underwater researchers are innovating underwater networks. In the following paper we will define a simulation environment that describes an underwater network where nodes can professionally surface the usage of radio communication. Using this environment, we will determine paths for radio communication on a line topology in a multimodal system. Numerous implementations are being done to simplify decisions about communication.

We will determine that distributed algorithms using only local information can perform at or near the level globally optimal algorithms for determining members of the radio path. This thesis contributes to the field of underwater networks by demonstrating the effectiveness of greedy routing schemes that use locally available information, and by providing a simulator in which they can be tested.

Underwater networks are a field that has been gathering attention. Land-based methods of network construction from discovery to advanced routing are all well established. Due to the unique constraints of operating in an underwater environment, many of these tried-and-true approaches need modification if they function at all. Peer discovery and low-level networking have been dealt with by previous research. In this paper we describe and evaluate eight possible routing schemes with different system-knowledge requirements. We will show that with a minimal set of information it is possible to achieve near-optimal results with energy costs considerably lower than centralized optimal algorithms. We demonstrate this by constructing and evaluating a custom simulation environment in MATLAB. This will be implemented in a mixed procedural and array-centric approach. Simulated networks are structured on a line topology. All nodes will be spaced along the horizontal axis at a random depth. It would be assumed that neighbor-discovery has been completed before the simulation starts, and all nodes have access to a global list of connected neighbors. We will demonstrate the effectiveness of distributed algorithms in an ideal environment, leading to the conclusion that near-optimal results can be achieved with local information only.

The agents emulated in this thesis will have depth control capabilities and two wireless modems. The first modem is an acoustic device that allows any two nodes within range to communicate underwater, but at a high energy cost. The second will be radio that cannot be used underwater but requires relatively little power. The depth adjustment system described in [1,2] consumes a large amount of power compared to either modem.

2. Related Work

It is common for sensor networks to rely on gateway nodes to handle large amounts of data over long ranges [3]. An example of this is the Sea Web [4] system. Sea Web used nodes at the surface to support communication with the outside world, and for localization within its own network via GPS. Research has been done that demonstrates the practical nature of using gateways in underwater networks [5]. Zhou et al. used linear programming to optimize the placement of these gateways to minimize power and delay [6]. What could be considered a flaw of these surface-gateway systems is that the acoustic modem limits communication. All nodes must transmit acoustically to

Corresponding author Email address*: shaharyar_abro@yahoo.com*

have data forwarded out of the underwater part of the network. This places a natural limit on the rate at which data can be extricated; in an ideal system data could be retrieved at the maximum rate of the modem, but a real system would face packet loss. To mitigate this problem, work has been done to introduce underwater vehicles to these systems and retrieve data as policy dictates.

These vehicles collect data along a path and occasionally rise to send data via radio [1,7]. The system we implement is different in that each node is capable of surfacing to send its own data or relay transmissions from its neighbors. Issues associated with acoustic gateways are removed in this scenario. The drawback of this approach is that nodes could be moved from their correct location for sensing, and the power-cost of rising needs to be factored in to routing decisions. For this paper we explore different approaches for determining radio paths on a line topology, attempting to minimizing energy consumption.

In [8] authors have worked on position adjustment–based position error–resilient geo-opportunistic direction-finding for void hole avoidance in underwater sensor networks. In [9] authors have done optimization of packet size for maximization in underwater Acoustic sensor networks. Researchers have also worked on the maintenance of underwater network sensors and multiple techniques are proposed for it [10].

3. Methodology

There are two main techniques for routing which are Acoustic algorithms and radio algorithms. Figure 1 shows the detailed types of algorithms and their sub algorithms.

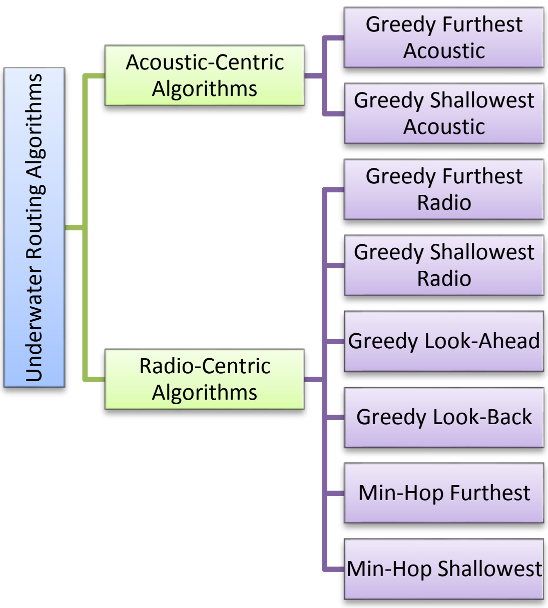

Figure. 1 Types of Underwater Routing Algorithms

An algorithm is well-thought-out to be acoustic-centric if routing choices are made from evidence on acoustic neighbors. So, ultimately, a radio-centric technique is one that makes choices based off radio neighbors.

Algorithm 1 Overview of Basic algo followed by all the techniques

```
loop
    receive radio packet P
    if P.Destination ≡ Self then
        exit
    else
        queue P for delayed-transmission
        find              neighbor  → For Every technique finding neighbor is
        send rise command             by different algorithm.
    end if
end loop
```

Algorithm 1 shows the overview of all the techniques, but every technique would adopt its own specific algorithm for finding the neighbor which is discussed in further paper. Below are discussed different types of underwater routing algorithms which are implemented in a simulation environment.

3.1 Greedy Furthest Acoustic

When a node receives a radio message it will use the connectivity matrix to determine its furthest connected neighbor. A rise command is sent to that neighbor, and after a short delay the radio message is re-broadcast. Growth commands are acoustic messages. They comprises no statistics; the arenas of the packet are terminus, source, and a protocol/command field corresponding to "rise". This process repeats until the radio message reaches its destination.

Algorithm 2 shows the flow being followed to determine the furthest connected neighbor.

Algorithm 2 Determining the furthest connected neighbor

```
function id = greedy_farthest_algorithm(self_ID,conn,dst_id,Node_Pos)
% pass in the destination ID, and the row of the present node
% in the connectivity matrix, and node positions
    id = 0;
    Conn_quick_ref = [];
    for i =1:length(conn)
        if i == self_ID;
            continue
        end
        if conn(i)
            Conn_quick_ref = [Conn_quick_ref i];
        end
    end
    Pos = zeros(length(Conn_quick_ref)+1,1);
    for i = 1:(length(Conn_quick_ref))
        Pos(i+1,:) = Node_Pos(Conn_quick_ref(i),1);
    end
    Pos(1,:) = Node_Pos(dst_id,1);
    dist = squareform(pdist(Pos));
    tmp_distance = inf;
    for i = 1:length(dist(:,1))
        if i == 1
            continue
        end
        if dist(i,1) < tmp_distance
            % if the distance is the least we've found
            % select that index and distance
            % that index can then be used to select from the quick ref.
            tmp_distance = dist(i,1);
            id = i;
        end
    end
    if id == 0
        error('Could not find any neighbors, which is not likely.\n')
    else
        id = Conn_quick_ref(id - 1);
    end
end
```

Below table shows the stats for this algorithm,

Table 1 Statistics for Greedy Furthest Acoustic

	Number of Nodes			
	25	**50**	**75**	**100**
Avg. Distance (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

3.2 Greedy Shallowest Acoustic

This approach requires nodes to determine their shallowest neighbor that is closer to the destination than themselves. The algorithm 3 depicts the algorithm it utilizes for determining the shallowest connected neighbor matrix.

Algorithm 3 Determining the shallowest connected neighbor

```
function id = greedy_shallowest_algorithm(self_ID ,conn ,dst_id ,Node_Pos)
    id = 0;
    Conn_quick_ref = [];
    for i =1:length(conn)
        if i == self_ID;
            continue
        end
        if conn(i)
            if i == dst_id
                id = i;
                return
            else
                Conn_quick_ref = [Conn_quick_ref i];
            end
        end
    end
    Pos = zeros(length(Conn_quick_ref)+2,3);
    for i = 1:(length(Conn_quick_ref))
        Pos(i+2,:) = Node_Pos(Conn_quick_ref(i),:);
    end
    Pos(1,:) = Node_Pos(dst_id ,:);
    Pos(2,:) = Node_Pos(self_ID ,:);
    dist = squareform(pdist(Pos(:,1:2)));
    depth_iter = -inf;
    for i=3:length(dist(1,:))
        if (dist(i,1) < dist(2,1)) && (depth_iter <= Pos(i,3))
            depth_iter = Pos(i,3);
            id = i;
        end
    end

    if id == 0
        error('Could not find any neighbors , which is not likely.\n')
    else
        id = Conn_quick_ref(id - 2);
    end
end
```

There is an immediate performance increase when selecting Greedy Shallowest Acoustic over Greedy Furthest Acoustic. As a rule, the energy cost of rising is greater than most acoustic messages. This algorithm will minimize the amount of travel required on a per-hop basis (average case) but is limited to selecting a neighbor within acoustic range.

Below table shows the stats for this algorithm,

Table 2 Statistics for Greedy Shallowest Acoustic

	Number of Nodes			
	25	**50**	**75**	**100**
Avg. Distance (m)	6.0429	5.8105	5.7679	5.8028
Avg. Depth (m)	10.1220	9.9952	9.9465	10.0011
Avg. Energy (J)	90.25	87.73	87.71	87.65
Avg. Time (μs)	63.2898	63.2063	62.2386	61.7403

3.3 Greedy Furthest Radio

The Greedy Furthest Radio algorithm determines its furthest connected radio neighbor and commands it to rise. It is functionally identical to Algorithms 1. Instead of using acoustic connection data, we instead use radio connections in Algorithm 2. Just like Greedy Furthest Acoustic, this ignores node depth and may or may not require multiple acoustic-hops for the rise command to reach its destination. This resembles a locally-optimal algorithm that tries to minimize the number of participating nodes without accounting for depth.

If all nodes are evenly spaced then this approximates a non-weighted minimum-hop approach. When nodes are spaced at or near the edge of acoustic or radio range is when this algorithm performs the worst.

As we would expect, by taking a basic approach and providing another layer of information, algorithm performance is increased. The power requirement is less than 3.1. his is solely due to the decreased number of nodes participating in routing. Fewer nodes are required to participate, so less power is used in moving.

Below table shows the stats for this algorithm,

Table 3 Statistics for Greedy Furthest Radio

	Number of Nodes			
	25	**50**	**75**	**100**
Avg. Distance (m)	2.9171	2.8147	2.6868	2.6114
Avg. Depth (m)	10.1220	9.9952	9.9465	10.0011
Avg. Energy(J)	45.00	41.48	40.19	39.74
Avg. Time (μs)	62.4871	63.4892	63.1409	63.0303

3.4 Greedy Shallowest Radio

Similar to Greedy Shallowest Acoustic, this algorithm identifies the shallowest of its radio neighbors, which may or may not also be an acoustic neighbor. This approach is identical to Algorithm 3; the relationship between the Greedy Furthest algorithms is the same as the relationship between the Greedy Shallowest algorithms. All of the best and worse cases are the same.

Below table shows the stats for this algorithm,

Table 4 Statistics for Greedy Shallowest Radio

	Number of Nodes			
	25	**50**	**75**	**100**
Avg. Distance (m)	2.4561	2.0758	1.9844	1.9707
Avg. Depth (m)	10.1220	9.9952	9.9465	10.0011
Avg. Energy(J)	36.03	31.72	30.36	30.07
Avg. Time (μs)	62.1548	62.7176	62.3132	62.2241

3.5 Greedy Look-Ahead

This is the first algorithm that requires the addition of neighbor's-neighbors positions. Greedy Look-Ahead utilizes linking info straddling out to the furthest radio neighbor's furthest neighbor.

It then computes the optimum next stage by means of Dijkstra's algorithm with link-costs existence biased by node deepness. The only necessities positioned on the scheme are: no-backwards traversals (mails continuously advance) and the track selected essentially have a length larger than dual (not including the sender). The track measurement condition is a safe-guard contrary to fetching a decently shallowest-neighbor method. The minimum weight path is selected, and assists as an estimate of a globally optimum route.

Below is the flow diagram for Dijkstra's algorithm.

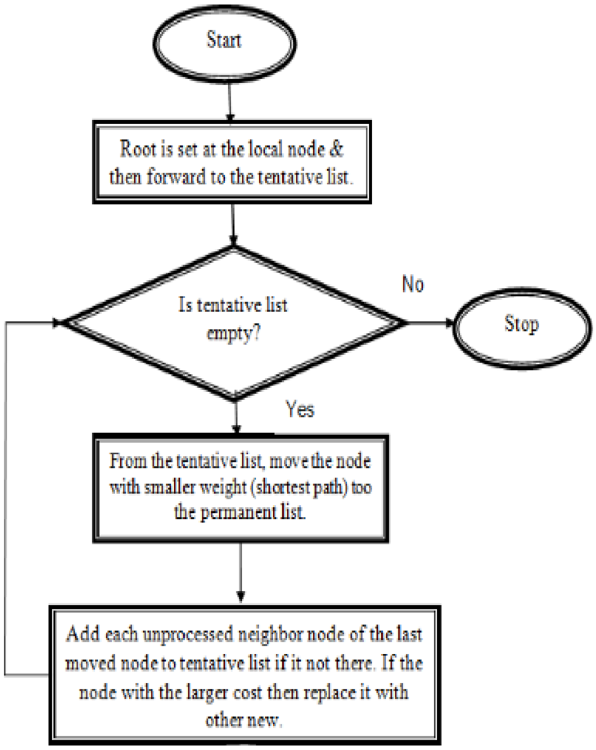

Figure 2 Dijkstra's algorithm

Below table shows the stats for this algorithm,

Table 5 Statistics for Greedy Look-Ahead

	Number of Nodes			
	25	50	75	100
Avg. Distance (m)	2.2097	1.8510	1.7401	1.6957
Avg. Depth (m)	10.1220	9.9952	9.9465	10.0011
Avg. Energy(J)	32.57	27.86	26.27	25.97
Avg. Time (μs)	79.3019	74.5612	68.6739	63.0941

3.6 Greedy Look-Back

Greedy Look-Back could be described as a time saving version of Greedy Look-Ahead. This algorithm sends a rise command to the furthest radio neighbor. On receiving of a rise knowledge, the receiving node will checkered to realize either it was the shallowest neighbor by the sending node. If so, it will rise and the algorithm continues onward. If not, it will appeal the look-ahead process against the unique sender, regulate the suitable node, and send a "forced-rise" command.

Algorithm 4 discusses the methodology for finding locally optimal forward moving paths.

Algorithm 4 Finding locally-optimal, forward-moving paths

```
function id = greedy_look_back_algorithm(self_ID, dst_ID, requester_ID, Positions, Connections)
    immediateNeighbors = [];

    if dst_ID == self_ID
        id = self_ID;
        return
    end

    for i=(requester_ID+1):self_ID
        if Connections(self_ID, i) && (Positions(i,3) > Positions(self_ID,3))
            immediateNeighbors = [immediateNeighbors i];
        end
    end

    if isempty(immediateNeighbors)
        id = self_ID;
        return
    end
    id = greedy_look_ahead_algorithm(requester_ID, dst_ID, Positions, Connections);
end
```

Below table shows the stats for this algorithm,

Table 6 Statistics for Greedy Look-Back

	Number of Nodes			
	25	50	75	100
Avg. Distance (m)	2.2097	1.8510	1.7401	1.6957
Avg. Depth (m)	10.1220	9.9952	9.9465	10.0011
Avg. Energy(J)	32.57	27.86	26.27	25.97
Avg. Time (μs)	64.9102	67.6236	66.8615	66.5082

3.7 Min-Hop Furthest

This is a globally optimal algorithm that does not take node depth into account. Using global position information, Dijkstra's algorithm is run with uniform costs applied to all network links. The best case scenario is an improvement over Greedy Furthest Radio. The worst case occurs when the shortest possible path includes nodes deeper than would be selected by Greedy Furthest. When this happens, we see a large increase in energy consumption. It is not a common case, but it is worth being aware of. The average case is on par with Greedy Furthest Radio. reason for the increase in run time is computational complexity (Dijkstra's algorithm). Every single message generated needed to be transmitted from the first node, to its destination. This includes rise packets. The power cost is elevated so greatly because the closer a node is to the origin, the more it is required to route. This is the same sort of issue that would be seen in a standard sensor network where all nodes share a common sink. Being close to the origin results in an uneven load on the system, which could be mitigated by distributing the algorithm or by including next-hops as data in a custom rise command.
Below table shows the stats for this algorithm,

Table 7 Statistics for Min-Hop Furthest

	Number of Nodes			
	25	50	75	100
Avg. Distance (m)	3.0701	2.7676	2.6652	2.6415
Avg. Depth (m)	10.1220	9.9952	9.9465	10.0011
Avg. Energy(J)	44.44	41.58	40.66	39.95
Avg. Time (μs)	65.6124	74.9882	80.8565	87.6294

3.8 Min-Hop Shallowest.

Using the global connectivity matrix and weighing links in the matrix by node depth, Dijkstra's algorithm determines the optimal weighted path from start to finish, consistently guaranteeing the least node movement. There is an excessive energy burden based on the algorithm being centralized. Ignoring that component, we should turn our attention to distance travelled and time. The average distances travelled are lower than any other algorithm. If only information on immediate neighbors can be known, the clear choice must be Greedy Shallowest Radio. If we allow for information on our neighbor's neighbors, the decision between Greedy Shallowest Radio and Greedy Look-Ahead depends on the cost of computation. The former is a close approximation of the latter, which is a close approximation of the ideal. For sake of simplicity, Greedy Shallowest Radio likely remains the strongest candidate. Below table shows the stats regarding this algorithm

Table 8 Statistics for Min-Hop Shallowest

	Number of Nodes			
	25	**50**	**75**	**100**
Avg. Distance (m)	2.1485	1.8068	1.6963	1.6591
Avg. Depth (m)	10.1220	9.9952	9.9465	10.0011
Avg. Energy(J)	31.89	27.21	25.66	25.32
Avg. Time (μs)	85.9095	86.8690	85.1852	84.1778

5. Results

Figure 3 shows the average amount of motion any given node could expect to move, at a given network size. The error bars on the plots represent standard deviation, and are only shown for algorithms that are significantly different from each other. The upper plot contains data on the acoustic-centric algorithms, with Greedy Shallowest Acoustic showing estimates on error.

The lower plot contains results on all but two radio-centric algorithms (excluded are Greedy Look-Back and Min-Hop Furthest). Both Greedy Shallowest Radio and Min-Hop Shallowest show error margins. We choose to separate acoustic and radio algorithms because they are effectively on different scales. The acoustic algorithms require more motion, and the standard deviation for the acoustic algorithms is considerably larger than for the radio algorithms.

Four algorithms are selected to have their energy use shown in Figure 4. They are Greedy Shallowest Acoustic, Greedy Shallowest Radio, Greedy Look-Ahead, and Min-Hop Shallowest. We chose to present an acoustic algorithm to show the vast discrepancy between acoustic and radio-based algorithms, and the radio algorithms shown will continue

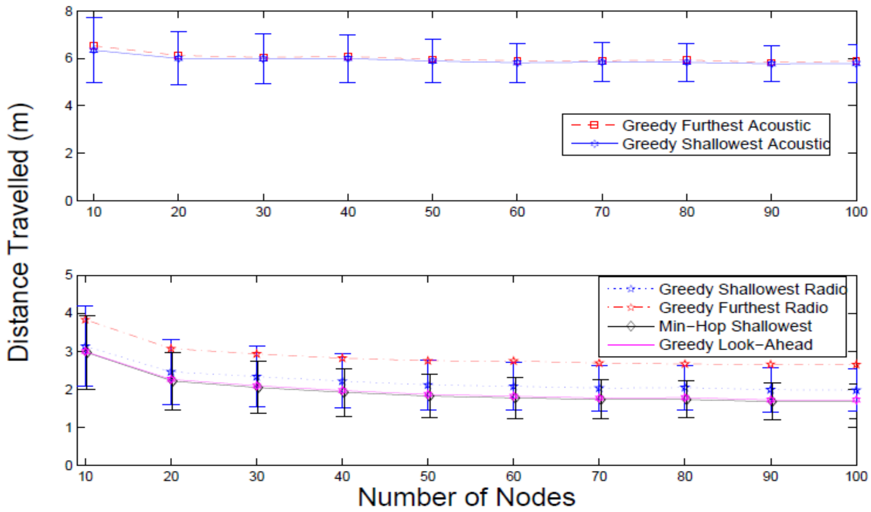

Figure 3 Graph of Start and End positions for Greedy Furthest Acoustic

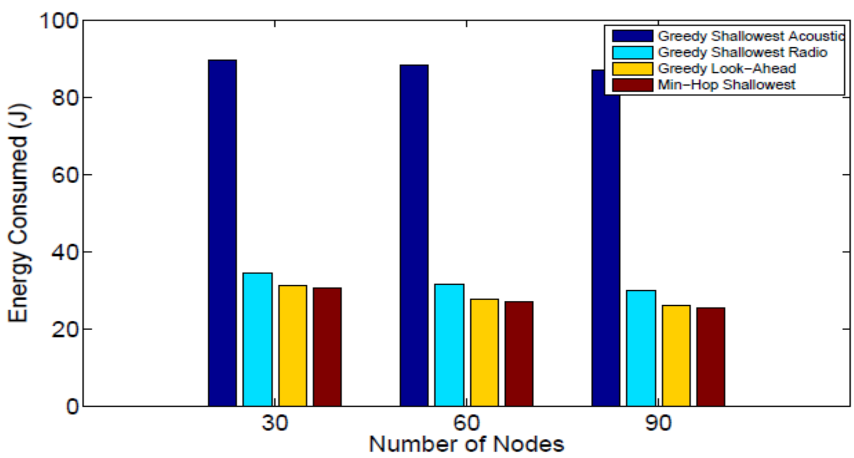

Figure 4 Graph of Energy Consumed by Four Algorithms

5. Conclusion

The goal of this paper has been to develop a simulation environment and a sample set of algorithms for evaluation. It is our sincere hope that this tool will be of use to researchers working on wireless underwater networks. There are many new and exciting approaches to the problem's aquatic systems face, and we hope we've made the process of exploring these approaches a little easier. It is critical we properly develop and leverage aquatic technology, so we can understand and make use of what the ocean provides.

References

[1] C. Detweiler, Decentralized Sensor Placement and Mobile Localization on an Underwater Sensor Network with Depth Adjustment Capabilities. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, July 2010.

[2] C. Chen and J. Ma, "Mobile enabled large scale wireless sensor networks," in Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference, vol. 1, p. 333338, 2006.

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

[4] J.-H. Cui, J. Kong, M. Gerla, and S. Zhou, "The challenges of building mobile underwater wireless networks for aquatic applications," Network, IEEE, vol. 20, p. 1218, June 2006.

[5] Z. Zhou, H. Yan, S. Ibrahim, J.-H. Cui, Z. Shi, and R. Ammar, "Enhancing underwater acoustic sensor networks using surface radios: Issues, challenges and solutions," in Sensor Networks (G. Ferrari, ed.), Signals and Communication Technology, p. 283307, Springer Berlin Heidelberg, 2009.

[6] B. Chen, P. C. Hickey, and D. Pompili, "Trajectory-aware communication solution for underwater gliders using WHOI micro-modems," in Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on, p. 19, 2010.

[7] Shah, Mehreen, Zahid Wadud, Arshad Sher, Mahmood Ashraf, Zahoor Ali Khan, and Nadeem Javaid. "Position adjustment–based location error–resilient geo-opportunistic routing for void hole avoidance in underwater sensor networks." Concurrency and Computation: Practice and Experience 30, no. 21 (2018): e4772.

[8] Yildiz, Huseyin Ugur, Vehbi Cagri Gungor, and Bulent Tavli. "Packet Size Optimization for Lifetime Maximization in Underwater Acoustic Sensor Networks." IEEE Transactions on Industrial Informatics (2018).

[9] Jambunathan, S., and P. R. Sowndariya. "Intelligent Routing Strategy Maintenance over Underwater Wireless Sensor Networks." *Research Journal of Advanced Engineering and Science* 3, no. 1 (2018): 242-246.

About Authors

Shaharyar Ahmed Abro completed his B.E in Computer Systems Engineering from Mehran University of Engineering & Technology, Jamshoro, Pakistan. Currently, He is enrolled in Master of Engineering from Computer Information Engineering (CIE), and going to be finished. He has been working as a Web developer and technical Writer. His research interests are Networking, design & development and Artificial Intelligence.

.

Prof. Dr. Tariq Jamil Saifullah Khanzada persuaded his Doctorate Degree the field of Wireless Information and Communication Systems from University of Magdeburg, Germany. **Currently,** working as a Professor at department of Computer Systems Engineering, Mehran UET Jamshoro Pakistan. His research interests are Wireless Indoor Positioning Systems, Signal Processing for Wireless Communications, Digital Communication Systems, Data Mining/ Data Sciences.

Shahbaz Wahab received his B.E in Computer Systems Engineering from Mehran University of Engineering & Technology, Jamshoro, Pakistan. Currently, He is currently enrolled in Master of Engineering from Institute of Information & communication Technology (IICT), and going to be finished. He has been working as a designer, developer and Computer Networks. His research interests are Networking, design & development and Artificial Intelligence.

Muhammad Mansoor completed bachelor of Engineering in Computer Systems Engineering from Mehran University of Engineering & Technology, Jamshoro, Pakistan. Currently, He is enrolled in Master of Engineering and going to be finished. He has been working as a designer, developer and technical writer in a Software House. His research interests are Networking, design and development and Artificial Intelligence.