1-1-2013

# Reactive localisation in underwater wireless sensor networks with self-healing

Mohamed Watfa
*University of Wollongong*, mwatfa@uow.edu.au

Tala Tsouli
*American University of Beirut*

Maya Ei Ayach
*American University of Beirut*

Omar Ei Ayach
*American University of Beirut*

# Reactive Localization in Underwater Wireless Sensor Networks with Self Healing

Mohamed K. Watfa
Faculty of Engineering
University of Wollongong in Dubai (UOWD)
Dubai, UAE
MohamedWatfa@uowdubai.ac.ae

Tala Nsouli, Maya Al-Ayache, Omar Ayyash
Computer Science/Computer Engineering Department
American University of Beirut
Beirut, Lebanon
Mma,tkn02,@aub.edu.lb

***Abstract*— In this paper, we present a novel technique for localizing an event of interest in an underwater environment monitored by an underwater sensor network. The network consists of randomly deployed identical sensor nodes. Instead of proactively localizing every single node in the network as all proposed techniques set out to do, we approach localization from a reactive angle. We reduce the localization problem to the problem of finding 4-Node Coverage, in which we form a subset of nodes such that every node in the original set is covered by four nodes belonging to this special subset – which we call the anchor nodes for simplicity. This subset of anchor nodes behaves like a backbone to the localization process. The anchor nodes are localized based on an "underwater GPS" preexisting system. Whenever a node detects an event, it is reactively localized using the anchor nodes, and the sink is supplied with the necessary information. By limiting the sensing range of the sensor nodes, once we have obtained the location of the node that has detected the event, we have a rough estimation of the location of the event. We show that in terms of energy consumption, this localization technique far surpasses others in terms of energy efficiency.**

***Keywords- localization; reactive localization; underwater sensor networks; enery-efficiency; k-node coverage***

## I. INTRODUCTION AND MOTIVATION

During the past few years, a significant interest in monitoring aquatic environments has emerged. Such a process was driven by major incentives such as scientific exploration, commercial exploitation, and coastline protection. These functions were made feasible by applying underwater communications among underwater devices. Underwater wireless sensor networks comprise a number of sensor nodes and vehicles installed at different levels of the ocean (surface, bottom, and mid-ocean) to perform various functionalities, most importantly monitoring the ocean environment. The underwater devices are connected via wireless links founded on acoustic communications. Underwater sensor nodes thus facilitated such applications as ocean bottom data collection, offshore discovery, disaster avoidance, pollution monitoring, navigation and surveillance.

These sensor networks share some properties with ground sensor networks, most notably the large number of nodes and the limited energy constraint, which poses a challenge in deploying and managing large scale wireless sensor networks. However the approach varies due to limitations posed by the underwater environment rendering a number of restrictions to the capabilities of the underwater sensor networks. Such limitations include, but are not restricted to, variable delays and limited bandwidth capacity.

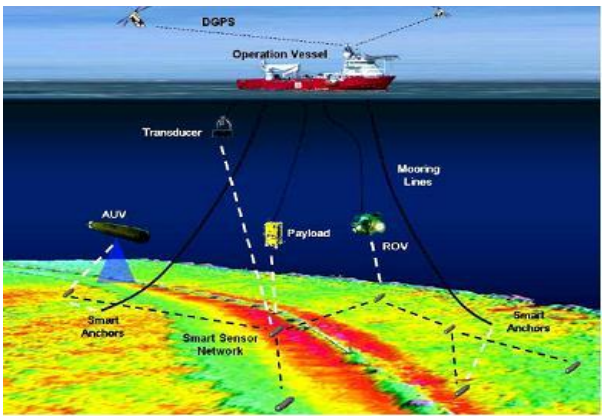


**Figure 1**: Typical Underwater Setting[1]

Figure 1 represents a typical setting for an underwater wireless sensor network. Smart sensors are anchored to the bottom of the ocean, and collaborate with Autonomous Underwater Vehicles (AUV) which navigate the ocean autonomously according to a set of instructions, and with Remotely Operated Vehicles (ROV) which are controlled from the Operation Vessel floating on the surface of water. The sensors and anchors cooperate to measure parameters and provide accurate position references to the AUV while they survey the ocean environment.

Wireless sensor networks have emerged as a fast-growing technology that has enabled us to monitor physical and environmental states like temperature, sound, vibration and pressure. As of yet these valuable sensor networks have been bound to terrestrial terrains; however, seventy percent of the Earth's surface is covered by enormous bodies of water: oceans, seas and rivers. Most of this territory remains unchartered, and this primarily gave a rise to the implementation of these sensor networks underwater.

[1] V. Chandrasekhar, W. K. Seah, Y. S. Choo, and H. V. Ee, "Localization in underwater sensor networks - survey and challenges," in Proceedings of ACM WUWNet'06, Los angeles, CA.

With wireless sensor networks, whether underwater or terrestrial, localization will inevitably be discussed. The importance of localization takes shape in the fact that much of the data obtained through these sensor networks must be location-aware. For example, imagine randomly deploying hundreds of nodes over a region with town A, town B and town C. The sensor with ID 98A returns a temperature of 23oC. This information is insignificant without a specific location or even a rough estimation of a location. Is the temperature 23oC in town A, B or C? This same logic applies to underwater sensor networks, where hundreds of nodes are deployed in a three-dimensional environment, and the tasks of these nodes are most often location-sensitive like pollution-tracking, military surveillance and animal tracking. Moreover, location data supports network layer services like geographic routing, clustering and topology control. However, for many applications, localization alone may be a poor choice of going about gathering data.

Like terrestrial wireless sensor networks, underwater sensor networks face many challenges – some of which they share with terrestrial sensor networks. Limited energy is a challenge common to both types of sensor networks as the deployed nodes cannot be reached for recharging. Another common challenge is multipath, where in UWSNs inter-symbol interference occurs, distorting the waves and hence leading to the loss of data.

All the localization schemes designed for underwater sensor networks (detailed in the Related Work section) handle the localization problem proactively. That is localization of all the nodes in the network is performed as a kind of initialization phase, meaning before the network is put to its actual use. However, if we study the motivation behind localization, we find that it is not necessary to know the location of every node in our network, since our aim is to localize an event of interest, rather than the node itself. Keeping that aim in mind, we notice that the energy expenditure incurred by a proactive localization algorithm is an unnecessary cost, and can be reduced by rendering the localization event-driven. That is we devise an energy-efficient reactive localization scheme.

Common to all underwater sensor network applications are the following challenges as depicted in Figure 2. First and foremost since radio frequency waves do no propagate well underwater, UWSNs resort to acoustic waves for communication. Acoustic waves are five times slower than RF waves, magnifying the propagation delay in UWSNs. Moreover, the speed of sound underwater is variable, being a complex function of temperature, pressure and salinity. Also the three-dimensional vast underwater environment poses a great challenge. The underwater environment imposes harsh physical limitations on the sensor networks which can be summarized by large propagation delay, low bandwidth capacity and high bit error rate. The underwater environment is also governed by currents and wildlife which poses the problem of node mobility as well as the problem of interfering noise – both man-made and ambient. Moreover, underwater sensor networks are challenged by two types of path loss. The first is attenuation, which is provoked by absorption of the acoustic waves, their conversion into heat, scattering, reverberation, refraction and dispersion. The second is geometric spreading, which is best described as the spreading of sound energy due to expansion of wave fronts.

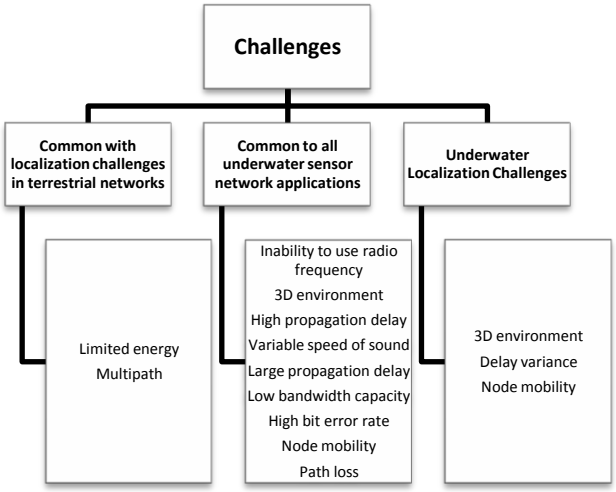


**Figure 2**: Challenges associated with underwater localization

The challenges that most affect localization in UWSNs are mainly the three-dimensional environment, which imposes a third dimension (unknown) to be determined by the algorithm. This calls for extra resources to make localization possible. For example, triangulation will need at least four non-coplanar nodes. Another challenge specific to localization is that the high delay in UWSNs is paired with a delay variance that makes the computation of RTT inaccurate (and hence its use not so effective). Node mobility also poses a great challenge for localization algorithms, since the nodes are almost constantly changing positions, implying that the localization algorithms must be able to either keep up or consistently refresh to keep track of positions. In our scheme, we reactively localize a node that detects an event, using a previously selected subset of anchor nodes with known positions.

The remaining sections of this paper are organized as follows. Section II provides an extensive overview of related work as compared to the technique we propose. In Section III, we elaborate on the details of our technique's function and architecture. Section IV provides an evaluation of the Reactive Localization Scheme in terms of energy efficiency, communication and computation overhead, storage overhead and localization error and coverage. Section V concludes this paper with a summary of the work done and an outline of future work.

## II. RELATED WORK

In this section, we present comprehensive overviews of localization schemes tailored for underwater sensor networks. The existing schemes differ significantly from the method we will propose in that all of them are proactive localization techniques. However, we rely on some of the methods presented in these schemes to achieve our setting and implement our algorithm.

Localization of sensor nodes in terrestrial environments has been widely explored in the past. The schemes proposed can be classified under two approaches, direct approaches and indirect approaches as depicted in Figure 3. Direct approaches, such as

GPS-based localization, involve absolute localization which does not particularly apply in underwater environments since such approaches are neither practical nor scalable nor adapt well with node mobility [1]. Indirect approaches are known as relative localization, since nodes position themselves with respect to their neighboring nodes. Commonly indirect approaches entail a small subset of nodes knowing their locations (via GPS), sending location information to neighboring sensor nodes, thus allowing them to calculate their relative locations. The localization process within the indirect approach can be classified into range-based localization and range-free localization. Range-based protocols provide more accurate location estimates as they use absolute point-to-point estimates; however they need additional complex hardware capacity thus increasing the cost. Range-free schemes are more cost-effective but provide less accurate location estimates. Range-based schemes are potentially good choices for underwater sensor networks.

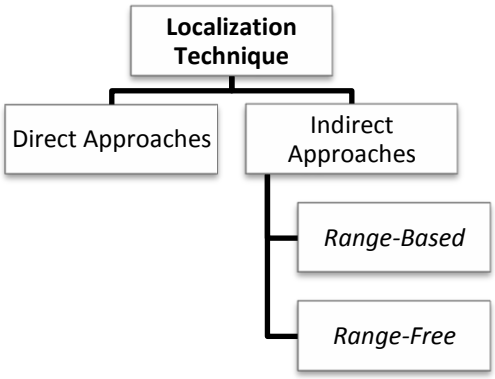


**Figure 3**: Classification of Localization techniques

Terrestrial localization has been widely investigated, but due to the several challenges posed by the underwater environment, common algorithms cannot be directly applied underwater. And thus in the recent years, authors have proposed localization schemes for small-scale underwater static networks such as [2], [3], [4], [5]. Some of these schemes use surface buoys and one hop communications between sensor nodes, such as GIB (GPS Intelligent Buoys) [3], and PARADIGM [2]. GIB is an "underwater GPS" system that relies on a centralized server to compute location information for nodes. PARADIGM involves autonomous underwater vehicles (AUV) computing their locations on-board.

Another scheme that uses AUV is presented in [11]. Erol et al. of "AUV-Aided Localization for Underwater Sensor Networks" [11] present a localization scheme for underwater sensor networks based on the use of an AUV (Automated Underwater Vehicle) that probes the underwater sensor field and assists nodes in calculating their coordinates. The proposed scheme assumes no initial infrastructure or synchronization between the nodes. Calculations and estimations gathered while the AUV is in motion result in significantly erroneous measures.

Hahn et al. in [7] put forward a centralized scheme that involves a sensor interrogating multiple surface buoys. It entails a ping-pong style that measures the round-trip delay for estimating ranges. The communication with surface buoys leads to network throughput degradation since the application messaging and the localization information share the same underwater channel. Opposite to that, our reactive scheme is keen on balancing an efficient communication overhead.

In [6], a silent positioning scheme is proposed where sensor nodes learn their locations by passively listening to beacon messages being delivered between neighbors. However in this scheme and contrary to our proposed one, it is not certain that we have four anchor nodes covering the node to be localized.

Contrary to our range-based scheme, [8] present ALS, which is area-based and range-free. It relies on the deployment of special anchor nodes that are capable of adjusting their power levels to divide a two-dimensional region into sub regions. Every anchor node has its own non-overlapping partition, and each sensor node collects its position estimate from a central server after providing all of the regions (for each anchor node) that it resides in. A major advantage distinguishes our scheme from ALS. Our localization system is characterized by a finer position granularity than ALS. That is to say, the positions of the sensor nodes obtained are within a coordinate system rather than positions within a sub region.

Additionally, Zhou et al. of "Localization for Large-Scale Underwater Sensor Networks" [10] propose a localization scheme that approaches the problem in a range-based hierarchical manner. The process is divided into two sub-processes: anchor node localization and ordinary node localization. They tackle this by integrating a three-dimensional Euclidean distance estimation method and a recursive location estimation method. Even though Euclidean estimation reveals to perform best in anisotropic topologies, it is hindered by its large computation and communication overheads. Anchor node localization is achieved through relying on surface buoys equipped with GPS sensors. The anchor nodes localize themselves based on the "underwater GPS" scheme, GIB (GPS Intelligent Buoys) [3]. This scheme is hindered by disregard to energy constraints and high communication overhead since it adapts continuous message flooding. It also entails higher deployment cost since it relies on a relatively big number of anchor nodes.

Zhang et al. [5] proposes UR-PLACE, a distributed protocol for underwater robot self-positioning that employs beacon flooding, multi-hop underwater robot networks, and iterative multilateral methods applicable only to small-scale static underwater networks since its high communication cost and low convergence speed make it inefficient for use in large-scale environments. [5] replaces the extensive local communication in [10] with global flooding, which essentially leads to the same bandwidth intensive usage which ultimately degrades the throughput.

In [12], Othman et al. proposes an anchor-free localization method for UWSNs. Their protocol performs node discovery and calculates relative locations. Node discovery begins with a primary seed node in a known position, which determines the relative positions of its neighboring nodes, and eventually other nodes in the network. The node discovery protocol prior to localization involves an immense amount of message exchange.

A new approach to the underwater localization problem is posed by Z. Zhou in SLMP [13] where mobility is taken into consideration. As Zhou's previously proposed schemes, SLMP is hierarchical process divided into two phases: anchor node localization and ordinary node localization. Every node performs future mobility predictions based on its past known location information, allowing it to estimate its future locations based on its predicted mobility pattern. The mobility predictions are prone to failure due to the random and sudden nature of many underwater movements (tides, animal interference, ships, etc…).

Yet another localization scheme for sparse 3D environments [14] transforms the three dimensional problem into a two dimensional one using projection techniques. Upon that, the authors design a purely distributed localization framework that can be applied with any ranging method proposed for 2D terrestrial sensor networks. The scheme incurs great storage and computation overheads, and also poses a problem of accumulating errors that total up to give erroneous node locations.

TABLE I. RELATED WORK AND THEIR MAJOR DISADVANTAGES

| **Localization Scheme** ***(Sorted by date of publication)*** | **Major Disadvantage** |
|---|---|
| Silent positioning in underwater acoustic sensor networks *(May 2008)* | Does not ensure that a sensor node is covered by 4 nodes (localizable) |
| SLMP *(April 2008)* | Based on mobility predictions that might miserably fail due to sudden changes in the underwater environment (tides, animal movements, etc…) |
| Underwater Localization in Sparse 3D Acoustic Sensor Networks *(April 2008)* | Accumulates error, great storage and computation overheads |
| AUV-Aided Localization for Underwater Sensor Networks *(Aug 2007)* | Erroneous measures due to motion of AUV |
| ALS *(May 2007)* | Anchor nodes must have extra capabilities and positions obtained are relative within a sub region |
| Localization for Large-Scale Underwater Sensor Networks *(Dec 2006)* | Large computation and communication overheads, disregard to energy constraints, and higher deployment cost |
| Node discovery protocol and localization for distributed underwater acoustic networks *(2006)* | Immense message exchange |
| Undersea navigation via a distributed acoustic communication network *(2005)* | Large overhead and network throughput degradation |
| A distributed protocol for multi-hop underwater robot positioning *(Aug 2004)* | Small-scale and static environment, low convergence speed, global flooding, bandwidth intensive, throughput degradation |
| GIB *(2001)* | Centralized and small-scale |
| PARADIGM *(2000)* | Small-scale and Erroneous measures due to motion of AUV |

## III. REACTIVE LOCALIZATION

In this section, we propose a scalable localization scheme for three dimensional underwater sensor networks. First, we present the underwater environment architecture we consider in our scheme. Then, we detail the function of the *Reactive Localization Process.*

### A. *The Architecture*

The architecture (see Figure 4) in which the Reactive Localization algorithm will apply is one equipped with two types of nodes. The sensor nodes and the surface buoys. Sensor nodes are randomly deployed over the desired area such that we assume that nodes will randomly sink to different depths depending on their densities. The nodes are therefore randomly deployed in the three dimensional environment. After selecting a subset of nodes, we refer to them as anchor nodes. The surface buoys are equipped with GPS. The sink is located on the surface in a well-equipped station where information will be gathered and computation will be possible.

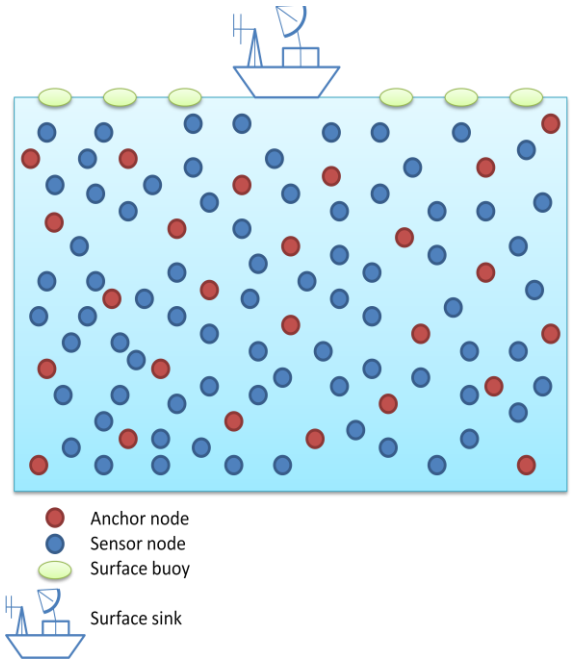


**Figure 4**: Underwater Network Architecture

We summarize in Table II some notations that would be used throughout the paper.

TABLE II. SYMBOL SIGNIFICANCES

| | |
|---|---|
| **SN** | Sensor node |
| **AN** | Anchor node |
| **D(x,y)** | Distance between x and y |
| **C($s_i$)** | Communication range of node $s_i$ |
| **CN(A)** | The set containing the sets of common neighbors between A and each of its neighbors |
| **R(Ai)** | The set of nodes that will replace Ai |

We detail three consecutive phases to solve the localization problem in an underwater 3D network:

1) Finding the anchor nodes
    a. Find a subset of nodes that provide 4-coverage
    b. Localize the anchor nodes
2) Reactive localization of sensor nodes
    a. A sensor node detects an event
    b. The sensor node localizes itself using the anchor nodes
3) Delivery of information
    a. Assuming a routing algorithm, the node transmits its location and information about the sensed event back to the sink

### *B. Finding the Anchor Nodes*

The first step is to find a subset of *anchor* nodes such that every sensor node is in the range of 4 anchor nodes.

#### *1) Problem Definition*

Every sensor node in the network must be covered by 4 non-coplanar anchor nodes.

$$\forall\ SN\ s_i \epsilon\ S\ \exists\ at\ least\ 4\ ANs\ b_j\ s.t\ D(s_i, b_j) \leq \min\ (C(s_i), C(b_j))$$

#### *2) Coverage Problem*

*Theorem 3.1:* In a k-1 dimensional environment, for a node to be localized, it must be covered by at least k nodes (k > 1).

*Proof:* As shown in Figure 5, three anchor nodes will only narrow down the choice of the location to two points. Having a fourth anchor node that is not coplanar with the first three, will make it possible to pinpoint the exact location of the sensor node in question.

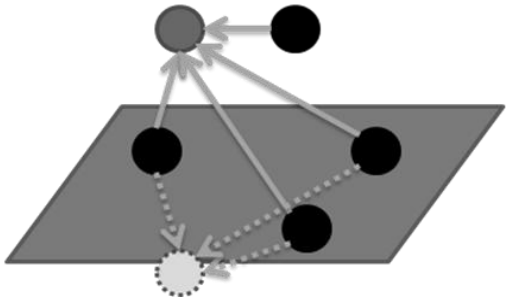

**Figure 5**: Importance of 4 anchors to localize a node in a 3D environment

■

Some points that we need to take into consideration are:

- There should exist at least D+1 anchors to uniquely localize a network in a D-dimensional space.
- To guarantee k Node-Coverage, each point should be within the sensing range of k or more sensor nodes.
- A 3D environment implies that we need 4 non-coplanar points

We elaborate on these points of Node-Coverage in order to rationalize the 4-Coverage Algorithm. We develop the idea of localizing a node in three dimensional space to solving for three unknowns (x, y, z). Mathematically, to be able to assign values to these three unknowns, we need four equations. The coverage algorithm guarantees 4-node coverage, which means that every sensor node in S should be within the communication range of 4 or more anchor nodes. The 4 anchor nodes, which are aware of their locations, will provide the sensor node attempting to localize itself with the needed four equations to solve for the three unknowns that will ultimately define its absolute location in the underwater medium. We will later provide a mathematical proof on how our proposed scheme effectively deals with the possibility that the four anchor nodes might be coplanar.

**Algorithm 1:** K-Node Coverage

1: *Send Hello Messages (ID, Energy)*
2: *Construct set of neighbors* $N_i$
3: *Broadcast set of neighbors* $N_i$
4: *Node waits for all neighbors to respond with sets*
5: **if** *node i receives 1 message with* $||N_j|| \leq k$, **then** *it is critical*
6: **if** *node i receives all messages with* $||Nj|| > k$, **then** *it can be turned off, sends REQUEST_TO_SLEEP message (after a time proportional to energy)*
7: *Nodes hearing the requests sends GO_TO_SLEEP to requester with lowest energy first*
8: *After receiving GO_TO_SLEEP from all neighbors, we send SLEEP and turns off*
9: *Step 7 for other requesters*

Our reactive localization scheme begins with an initialization process that determines a subset of nodes, called the anchor nodes, such that every sensor node (ordinary node) is covered by four anchor nodes. That is achieved by the K-Node Coverage Algorithm, in the case when k is set to be equal to 4. After randomly deploying the sensor nodes in the underwater environment, every node broadcasts a hello message with its ID number and energy level to its neighbors. Every node, upon receiving the hello messages from all of its neighbors, constructs a table of its neighbors, and then broadcasts that table to its neighbors. A node waits for time = $\tau$ till it receives the neighbor sets from all of its neighbors. At that point every node is aware of its neighbors and the neighbor set of each of it neighbors. If one of the sets received by a node is of a size equal to 4, then the receiving node is a critical node and cannot be turned off. If all of the sets received by the node are of size greater than 4, then the node may be turned off, and so it waits for a period of time inversely proportional to its energy level, and then broadcasts a REQUEST_TO_SLEEP message. By waiting for a period of time inversely proportional to energy level, we are giving nodes with the lowest energy level the priority of going into the sleep state. Nodes hearing the REQUEST_TO_SLEEP send a GO_TO_SLEEP message to the requester with the lowest energy level first. If a node receives a GO_TO_SLEEP from all its neighbors, it will broadcast a SLEEP message and goes into a sleep state. After the completion of that phase for all requesters, the nodes that remain awake are the chosen subset we shall refer to as anchor nodes, and the nodes in the sleep state are the sensor nodes.

*3) Localizing the Anchor Nodes*

After finding the subset (anchor nodes), we tackle the problem of localizing the chosen nodes. To localize the anchor nodes, we resort – as previously mentioned – to regarding anchor nodes as nodes that are capable of communicating with surface buoys and localizing themselves. We assume this property for all deployed nodes since the subset of anchor nodes is determined after deployment and thus no nodes are "special". Using existing underwater GPS systems, such as GIB [3], the anchor nodes with their ability to communicate with several surface buoys can localize themselves. Obviously, due to the complexity and energy consumption of GIB, it cannot be used on all the deployed nodes leading to our proposed research work.

### C. Reactive Localization of Sensor Nodes

After the anchor nodes are selected and localized, we outline the function of sensor nodes upon detecting an event. First a sensor node detects an event. The sensor node broadcasts a message to its one-hop neighbors, four of which will be acting as anchor nodes based on the *4-Coverage Algorithm*. The message broadcasted will be referred to as a *Localization Request Message*. Once the anchor nodes receive the messages, they reply with their location information. The node hence localizes itself, using this information, by *quadrilateration*.

We describe quadrilateration by briefly defining *multilateration.* Multilateration is a range-based localization scheme, in which the sensor node measures distances to anchors by time of flight (TOF). Mathematically, we need *n+1* (4) linearly independent equations to solve a system in *n* (3) dimensions. These four messages, sent from four different anchor nodes, will make four sets of coordinates available to the node, which it uses to solve the equations:

$$(x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2 = d_i^2$$

It follows from this definition of multilateration that *quadrilateration* is the localization process in which nodes measure distances from 4 reference points.

We will have two modes for sensor nodes: Localized and Non-Localized. Initially all sensor nodes (non-anchor nodes) have a Non-Localized state. These two states are governed by a timer. Once localized, a node will have a Localized status for a preset interval of time. When the time expires, the node discards its location and its status is once again Non-Localized. This process ensures that if a node that detects an event continues to detect it for a consecutive period of time, it will not have to localize itself several times.

To elongate the lives of the sensor nodes and conserve energy, we make it such that the sensor nodes have sleep/wakeup cycles. While asleep, the sensor nodes cannot communicate with each other but continue to sense the environment and try to detect events. Once an event is detected, the sensor node wakes up. Periodically, the sensor nodes wake up in case other sensor nodes are trying to contact them for self-healing. These sleep/wakeup cycles efficiently maintain energy levels and make it possible for the sensor nodes to function normally at the same time. Anchor nodes are always awake and listening for some sensor node that may attempt to contact them for localization information.

### D. Delivery of Information

The idea behind this algorithm is localizing a node that detects an event and thus obtaining a rough estimation of the event's location. It is understood in this scheme that several nodes may detect the same event. In this case, all of these nodes will send localization requests. The information from all of the nodes is sent back to the sink, where the messages are interpreted and a more accurate localization for the event is obtained. This part of the process can be seen as a range-free localization of the event.

For instance, if node A in Figure 6 detects Event Z and nodes B, C and D detect it as well, the sink will receive the locations of all nodes and their distances from Event Z. Based on this information, the location of the event can be narrowed down to a smaller area as shown. Hence the higher the node density, the more likely that many nodes will detect the same event and the more accurate the location of the event is.

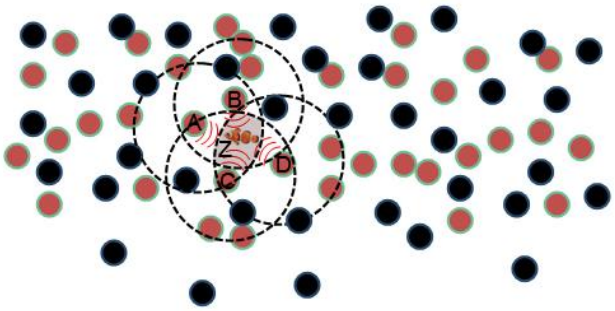


**Figure 6**: Several nodes detecting the same event and giving a more accurate localizat of the event

## IV. SELF HEALING

Since anchor nodes are indispensable to the localization function and they are likely to die faster than sensor nodes since they are being exhausted, we propose a self-healing algorithm that will find substitutes for a dying/dead node. This is done in order to avoid repeating the first step: finding a subset of anchor nodes. The self-healing algorithm we present next is divided into two parts to handle two possible situations in which self-healing algorithms are needed:

- Low Energy Self Healing
- Sudden Crash Self Healing

### A. Low Energy Self Healing

We wish to replace $A_i$, an anchor node whose energy-level is reaching its end. This algorithm is activated once a node "realizes" that its energy level has become low. We set an energy threshold $E_T$ that is chosen such that the anchor node will have enough energy to perform this operation before dying completely. When the node detects that its energy level has reached the threshold $E_T$, it commences with the Low-Energy Self-Healing algorithm.

**Algorithm 2:** Low-Energy Self-Healing

1: *Construct the neighbor set – not including anchor nodes – of* $A_i$. *Initially,* $CN(A_i) = \varphi$.
2: $A_i$ *broadcasts a message requesting that each of its non-anchor neighbor nodes to construct their neighbor sets, which include nodes within their communication range. The message also contains the nodes that* $A_i$ *covers. The neighbor nodes reply with a set of common neighbors between them and* $A_i$.
3: **for** *each neighboring node* $n_j$ *that receives the message and replies with its common neighbor set to* $A_i$ **do**
4: $CN(A_i) = CN(A_i) \cup n_j$
5: **end for**
6: **if** $CN(A_i) != \varphi$ **then**
7: *choose the largest set (i.e. with most common neighbors)*
8: $R(A_i) = R(A_i) \cup$ *Node providing largest set*
9: *form an intersection with the remaining neighbor sets and the remaining uncovered neighbors*
10: *Repeat from step 6 until* $CN(A_i) = \varphi$

The node constructs its neighbor set, comprised of all one-hope neighbors except for other anchor nodes. We do not want to choose other anchor nodes because they are already accounted for and will not participate in replacing our dying node. The anchor node we wish to replace sends a message to its one-hop neighbors, requesting that each of these neighbors construct their own neighbor sets. Along with this request, the anchor node also sends a list of the nodes it "covers". Every sensor node that receives this request constructs its own neighbor set as requested, and then finds the intersection between its neighbor set and the neighbor set of the dying anchor node. It sends this intersection back to the anchor node.

Having received all of its neighbors' sets, the anchor node proceeds to choose the largest set (covers most of its neighbors). This chosen set is reflected in the choice of the node responsible for it as one of the replacements of the dying node. The anchor node removes this set from the list of sets and removes the neighbors that have been covered by this set from the list of neighbors that remain uncovered. Then it finds the intersection of the list of remaining neighbors and the neighbor-sets provided by its neighbors. From these intersections it chooses the largest set again.

This process is repeated until no neighbors of the dying anchor node remain uncovered. The set of neighbors chosen will become anchor nodes and will proceed to localize themselves as described earlier.

### B. Sudden Crash Self Healing

Sudden crash self-healing is induced by the discovery that an anchor node A has suddenly crashed. The discovery occurs in the following manner. First sensor node S covered by A detects an event. S typically sends localization request messages to the anchor nodes that cover it. When S receives no response from anchor node A - which it knows to be amongst its 4 anchor nodes - S deduces that A is dead. S then begins the Sudden Crash Self-Healing Algorithm on anchor node A.

**Algorithm 3:** Sudden-Crash Self-Healing

1: *S sends to its non-anchor neighbors a* **Localize-Yourself Request**
2: *The neighbors all reply to S with whether or not they are able to do this*
3: *S then chooses one of the nodes and requests that this node localize itself. S may choose the node on adjustable criteria such as energy level or distance or RSS.*
4: *The node localizes itself as requested, and S uses this node to replace its dead* $4^{th}$ *reference point*

We can allow this algorithm to encompass as many hops as we wish. The number of hops indicates how far each node will propagate this *Localize-Yourself Request.* For example, if a node S that cannot localize itself due to a dead reference point, one of its neighbors is likely to share this dead reference point and will also be unable to localize itself. This neighbor can propagate the request until one of the neighbors *can* localize itself and act as a reference point. If a node that is not a one-hop neighbor of the original node S does localize itself in response to a request, then this localization will have to trail backwards until the original node has its $4^{th}$ reference point. It is much better however if the node that localizes itself is a one-hop neighbor of the node in question; since that way, we will minimize the delay before the node is localized and the detected information is sent back to the sink.

This is not an ultimate solution for sudden-crashes but it can be tailored to be the most efficient time and energy-wise. However in terms of maintaining the subset, it will not do that since anchor nodes normally act as a "$4^{th}$ reference point" for more than one node, and it is unlikely that when a node performs sudden crash self-healing that the $4^{th}$ node it eventually chooses will cover all of the nodes its dead predecessor covered. Hence this algorithm may have to be run several times for the death of the same node.

## V. THEORETICAL ANALYSIS

In this section, we provide in depth theoretical analysis and proofs of some of the stated theorems and assumptions.

### A. K-Node Coverage Localizing ALgorithim

*Theorem 5.1:* The probability that the 4 anchor nodes covering the sensor node all lie on the same plane, $P_{coplanar}$, is 0.

*Proof:* Since anchor nodes are not selected before hand, we have no special control on their deployment and thus locations. This poses problems, one of which is the probability of four anchor nodes involved in localizing a fifth node lying on the same plane. If the four nodes lie on the same plane, we cannot properly localize a fifth node using them. This case must be handled; we will do so by proving that this event's probability is zero.

Consider 3 points $A(x_A, y_A, z_A)$, $B(x_B, y_B, z_B)$, $C(x_C, y_C, z_C)$ of known positions and a $4^{th}$ point $D(x,y,z)$. The problem is proving $D \in \Delta ABC$. Although $D(x,y,z)$ might be correlated to the positions of A, B, C, we make no assumptions about this correlation. However, we can safely say that $x_D$, $y_D$, $z_D$ are logically independent and thus probabilistically independent.

Moreover, due to the many factors affecting current, drift, velocities, etc… we can assume that the nodes' distribution is sufficiently random (i.e. continuous and thus free of dirac deltas and probabilistic peculiarities).

To simplify the analysis, we will assume that the distribution of x, y, z are normal distributions. Let $\eta(\mu, \sigma^2)$ be the normal distribution with mean $\mu$ and variance $\sigma^2$

$$x \sim \eta(\mu_x, \sigma^2_x)$$
$$y \sim \eta(\mu_y, \sigma^2_y)$$
$$z \sim \eta(\mu_z, \sigma^2_z)$$

For A, B, C, D to be coplanar,

$$u_{\overrightarrow{AB}\times\overrightarrow{AC}} = u_{\overrightarrow{AB}\times\overrightarrow{AD}} \quad (1)$$

where $u_{\vec{v}}$ is the unit vector in the direction of $\vec{v}$

$\vec{n}$ = AB x AC (normal vector)

This means that,

$$\frac{\overrightarrow{AB}\times\overrightarrow{AC}}{\|\overrightarrow{AB}\times\overrightarrow{AC}\|} = \frac{\overrightarrow{AB}\times\overrightarrow{AD}}{\|\overrightarrow{AB}\times\overrightarrow{AD}\|} \quad (2)$$

$$\frac{\begin{pmatrix}x_n\\y_n\\z_n\end{pmatrix}}{\left\|\begin{pmatrix}x_n\\y_n\\z_n\end{pmatrix}\right\|} = \frac{\left(\begin{pmatrix}x_{AB}\\y_{AB}\\z_{AB}\end{pmatrix}\times\begin{pmatrix}x_D-x_A\\y_D-y_A\\z_D-z_A\end{pmatrix}\right)}{\|\overrightarrow{AB}\times\overrightarrow{AD}\|} \quad (3)$$

$$\frac{\begin{pmatrix}x_n\\y_n\\z_n\end{pmatrix}}{\left\|\begin{pmatrix}x_n\\y_n\\z_n\end{pmatrix}\right\|} = \frac{\begin{pmatrix}x_{n\prime}\\y_{n\prime}\\z_{n\prime}\end{pmatrix}}{\|\overrightarrow{AB}\times\overrightarrow{AD}\|} \quad (4)$$

Since $x_D$, $y_D$, $z_D$ are Gaussian random variables, then $x_D$-$x_A$, $y_D$-$y_A$, $z_D$-$z_A$ are also Gaussian. Moreover $x_{n'}$, $y_{n'}$, $z_{n'}$ are Gaussian (only manipulation with constants and elements are jointly Gaussian)

So $\begin{pmatrix}x_{n\prime}\\y_{n\prime}\\z_{n\prime}\end{pmatrix}$ is a Gaussian vector and thus $\left\|\begin{pmatrix}x_{n\prime}\\y_{n\prime}\\z_{n\prime}\end{pmatrix}\right\|$ has a Rayleigh distribution. The above vectors are unit vectors and thus for them to be equal they must have the same angles θ and ∅; However, it is proven that in such vectors, θ and ∅ have uniform distributions. So, the problem reduces to the probability of

$$\theta = \theta^* \text{ and } \varnothing = \varnothing^* \quad (4)$$
$$\theta_{n'} \quad \theta_n \quad \varnothing_{n'} \quad \varnothing_n$$

The probability of which is identically 0 (since continuous uniform distance). So, we can conclude that the probability that the 4 anchor nodes covering the sensor node all lie on the same plane, $P_{coplanar}$, is 0 ■

*Definition 5.1:* A node is critical when one of its neighbors needs it to be k-covered.

*Lemma 5.1:* Critical nodes are never turned off.

*Proof:* Consider a node $s_i$. The node $s_i$ waits for time τ to receive the set of neighbors $N_j$ from every neighbor $s_j$. Considering bidirectional links, $s_i$ will be an element of every $N_j$ received by $s_i$.

$s_i \in N_j$ iff $s_i$ received $N_j$ from $s_j$. We consider three cases:

- If for some j, $\|N_j\| < k$

  $s_j$ has less than k neighbors including $s_i$, and in that case $s_j$ cannot be localized, since it is not covered by k nodes. In that case $s_i$ is considered critical, and is kept turned on.

- If for some j, $\|N_j\| = k$

  $s_j$ has exactly k neighbors including $s_i$, and in that case $s_i$ is critical since to be localized, $s_j$ needs to be covered by k nodes. Since $s_j$ has exactly k neighbors, then each one of its neighbors is critical for it to be localized. In that case $s_i$ is considered critical, and it does not send a REQUEST_TO_SLEEP message, and hence remains in the awake phase.

- If for some j, $\|N_j\| > k$

  $s_j$ has more than k neighbors including $s_i$, and in that case any of $s_j$'s neighbor can be turned off since we only need k nodes covering it. In that case $s_i$ is not considered critical, and it broadcasts a REQUEST_TO_SLEEP message, which is received by all its neighbors. The node does not sleep yet, until it receives a GO_TO_SLEEP message from all of its neighbors.

Since the algorithm guarantees that only non-critical nodes send a REQUEST_TO_SLEEP message, then critical nodes are guaranteed to remain awake. ■

*Lemma 5.2:* Nodes must make a collective decision concerning which nodes can go to sleep.

*Proof:* We prove this lemma by contradiction. Assume the opposite: A node, upon finding out that all of its neighbors can be covered by k nodes, it directly goes to sleep.

Consider node $s_A$ and $N(s_A) = \{s_B, s_F, s_G, s_L, s_N, s_P\}$. Every node $s_i$ in $N(s_A)$ will receive this set, and calculate that $\|N(s_A)\| > 4$. Similarly, every node $s_i$ will receive similar sets from every neighbor $s_j$ such every $\|N(s_j)\| > 4$. If upon that event node $s_i$ can be turned off, then all the nodes $s_B$, $s_F$, $s_G$, $s_L$, $s_N$, $s_P$ will turn off $\Rightarrow N(s_A) = \varnothing$ and $s_A$ covered by 0 nodes and cannot be localized. Hence, the nodes must make a collective decision concerning which nodes can go to sleep. Therefore, a node can only turn off when it receives a GO_TO_SLEEP from all of its neighbors. ■

*Theorem 5.3:* The algorithm ensures that every node is k-covered if it initially had more than k neighbors.

*Proof:* If a node does not initially have more than k neighbors, all of its neighbors stay awake then it cannot be localized, according to *Theorem 1*. A node issues a REQUEST_TO_SLEEP if it determines that it is not critical for the localization of all its neighbors, i.e. all its neighbors have 4 other neighbors. After issuing this request, a node will

only sleep after it receives confirmation from its neighbors that they will indeed maintain a set of more than 4 active neighbors after it goes to sleep. This last step is to ensure that not all neighboring nodes of a node with initially more than 4 neighbors go to sleep simultaneously (Lemma 2). ■

*Theorem 5.4:* The communication complexity of K-Node Coverage is *O(nm)* where *n* is the total number of nodes deployed, and *m* is the maximum number of neighbors a node has.

*Proof:* During the initialization phase, a node sends at most one REQUEST_TO_SLEEP, at most one GO_TO_SLEEP request per neighbor and a maximum of one SLEEP message to be broadcasted to its neighbors. Hence each node sends a maximum of $O(m)$ messages, which means that the total message complexity is $O(nm)$. ■

### B. Self Healing Algorithim

#### 1) Run-Time of Low Energy Self-Healing

Let N be the number of neighbors for a certain node. In order to choose the largest set amongst the available sets, we need a linear search algorithm, which is of $O(N)$. This can be compared to choosing the maximum out of a list. We proceed to form the intersection of remaining uncovered neighbors and remaining sets, which is estimated at $O(N^2)$ time. The largest set will have to be chosen again. In the worst case scenario (probability of this scenario happening is negligible) this will happen N times so choosing the set will require $O(N^2)$. We deduce the overall time complexity of the algorithm at $O(2N^2) = \boldsymbol{O(N^2)}$.

#### 2) Run-Time of Sudden Crash Self-Healing

There is no real computation in the sudden-crash self-healing algorithm. The time it requires this algorithm to converge is simply the typical communication delays and the time required for a node to localize itself. After that, the node proceeds to localize itself as detailed before.

## VI. Analysis and Evaluation

In our simulation experiments, 500 sensor nodes are randomly distributed in a 100m x 100m x 100m region with 50 anchor nodes. We define node density as the expected number of nodes in a node's neighborhood; hence node density is equivalent to node degree. We control the node density by changing the communication range of each node while keeping the area the same. We study the differences as compared to other underwater localization schemes (mainly [7], [8], and [12]). Table I shows the typical energy consumption of each sensor node action where all nodes were initialized with an energy capacity of 1000 Joules. Finally, in order to obtain statistically significant results, we report average results of 10 simulations in each of the experiments carried out. The confidence level is 95%.

### A. Localization Coverage

Localization coverage is defined as the ratio of localizable nodes to the total number of nodes. Clearly, as can be seen from Figure 7, as node density increases, localization coverage increases. Once the nodes are dense enough so that the subset of anchor nodes can be sufficiently completed, then localization coverage will be at 100% and errors will be small. Since a complete set implies that the condition of every sensor node being covered by four anchor nodes is achieved and hence whenever a sensor node needs to be localized, it *can* be localized. In other words, every node is hence localizable. The percentage of coverage increases linearly as node density increases. It also increases as the subset grows to incorporate more anchor nodes. This implies that we may be able to overcome the low localization coverage in sparse networks by making our subset larger.

In comparison to the hybrid scheme and the recursive scheme [12] proposed in [11], ***our algorithm is slightly lower in terms of localization coverage with lower density; however, it quickly catches up to achieve the same results with more accuracy.*** We notice that the difference is not very big at the beginning because we choose our anchor nodes to optimally cover the nodes in the area, but the hybrid algorithm achieves slightly better coverage due to their use of recursion.

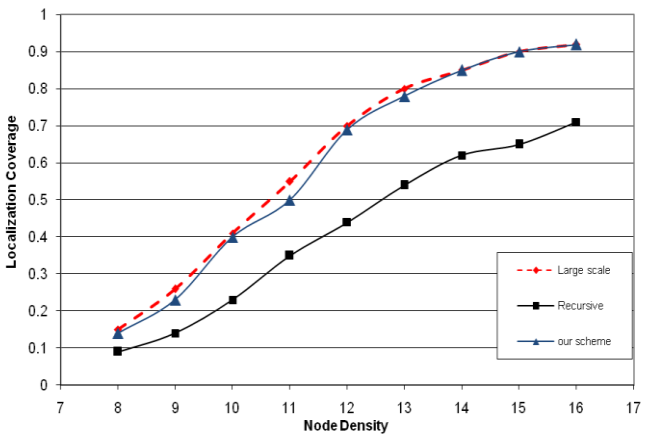


**Figure 7**: Localization coverage of different approaches.

### B. Localization Error

In order to localize a sensor node, a distance estimate to each of the four anchor nodes is acquired through Time-of-Flight measurements. Predictably, there will be a certain measure of error; however, this error does not accumulate since every node is localized from anchor nodes, the positions of which we assume to be devoid of errors. The positions of the anchor nodes are the most accurate relatively. The techniques used to localize the sensor nodes that detect the events are not new, and thus the error estimated in our scheme is similar to the error encountered by other localization algorithms. It is expected to encounter a lesser degree of error than that measured by [11]; since the error there recursively accumulates.

In general, as depicted in Figure 8, the localization error is higher when the nodes are sparse since the subset of nodes we choose may be lacking in the sense that a node may not have four other nodes that cover it. At higher density, the error

should resemble the error faced by other schemes. At a certain density that will provide what we have come to refer to as a "complete subset", the error will have reached a minimum beyond which it will no longer decrease no matter how the density increases. When self-healing occurs, we expect the error to increase since there will be an accumulation of errors to account for; however, we do not represent this in the graphs.

Compared to the hybrid scheme, we notice that our algorithm begins with a slightly higher percentage of error at lower density; however this quickly changes. And while error continues to decrease as we increase the node density in our algorithm, their error percentages are almost constant all throughout since the recursion in their algorithm leads to a propagation of error through the system. As for the AUV-Aided Scheme [10], we notice that their errors fluctuate and are hence unreliable since the error is dependent on a chosen interval for the AUV to transmit signals. ***For a higher node density, our algorithm far surpasses both in terms of accuracy.***

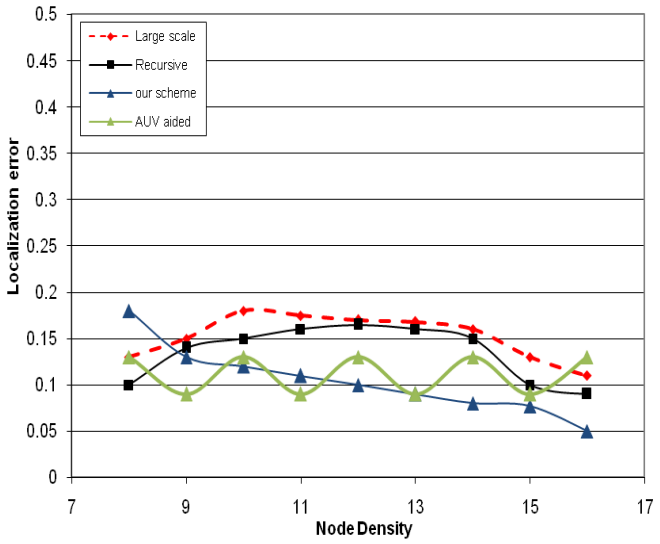


**Figure 8**: Localization error of different approaches.

## C. Storage and Computation Overhead

The storage overhead imposed by the Reactive Localization Algorithm is mostly due to the K-Node Coverage subroutine. As a result of that algorithm, we have every anchor node storing the list of nodes it covers and its status as an anchor node. All nodes will have to store $E_T$, the energy sthreshold – just in case they, too, become anchor nodes as a result of a self-healing algorithm. All sensor nodes will also have to store their locations once they are localized and their statuses (localized or non-localized). Like other metrics in this localization problem, the storage overhead varies with node density. As node density increases, the storage overhead decreases at first (since the anchor node subset will increase and hence each anchor node will be responsible for less sensor nodes); however, after the "subset completion" point mentioned before, the storage overhead to start increasing again; since every anchor node will be responsible for a greater number of sensor nodes.

Computation overhead is encountered mainly during the self-healing algorithms. We have already computed the runtime of the low-energy self-healing algorithm to be of order $n^2$ where n is the number of neighbors of the dying anchor node.

## D. Communication Overhead

We study the communication cost versus time in our algorithm and proceed to compare it to other algorithms by plotting the communication cost versus node density. The Reactive Localization Algorithm minimizes the communication overhead. While it may start out with high communication overhead, in order to select the subset and localize it, we consider this to be an initialization phase. After this phase, the curve slopes downwards until it reaches an average constant that is the cost of detecting an event and communicating with neighbors in order to be localized. Only upon the detection of an event does the algorithm require that the nodes communicate, and this communication is just with one-hop neighbors. Hence, we see why the curve starts out as slightly higher than other algorithms and then dies out as time progresses.

The communication overhead in the sudden-crash self-healing algorithm depends on the number of hops encompassed by the algorithm. The more hops, the higher the communication overhead. Otherwise, it is the cost of broadcasting a message to one-hop neighbors, receiving a reply and then selecting a node to localize itself. It also includes the communication overhead imposed by a typical sensor node localizing itself. We also study (in Figure 9) the communication overhead relative to node density as compared to the hybrid scheme. On average, our communication cost is less than their communication cost. Although we might start out with higher communication cost, our algorithm compensates as mentioned before by decreasing the communication cost after the initialization phase. Also on average the communication cost is higher on low node density since the nodes will continuously try to find a fourth reference point in order to localize themselves. Then, as the node density increases and the subset becomes more "complete", the communication cost decreases as there will be less need for self-healing algorithms.

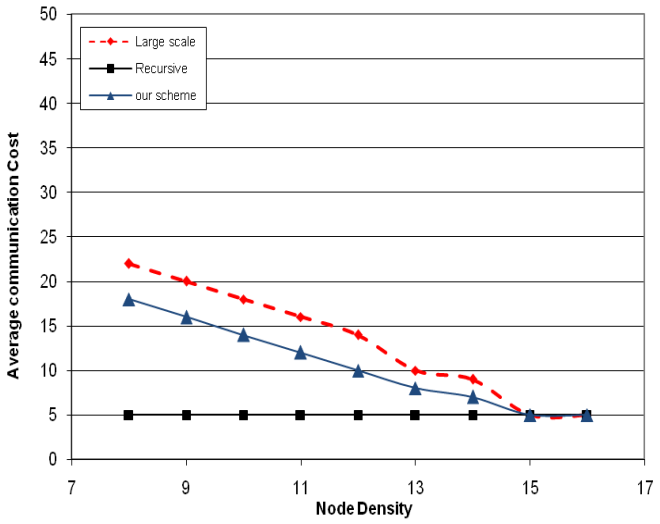


**Figure 9**: Communication cost of different approaches.

### *E. Energy Consumption*

Being reactive, one of the ultimate goals of this novel technique is to reduce energy consumption which it accomplishes effectively. After the selection of the subset, the nodes alternate sleep/wakeup cycles until they detect an event or until contacted by another node for a localization request in case of self-healing. In the initialization phase, we expect to consume more energy than other algorithms to set up the subset. We are not concerned with this energy expenditure, since the later phase in which sensor nodes are alternating between sleep/wakeup cycles will compensate for the extra energy expenditure by allowing the nodes a longer life as shown in Figure 10. We plot the node energy versus time in our algorithm and in a typical proactive localization algorithm, the hybrid scheme of [11]. The curves clearly show what our analysis says.

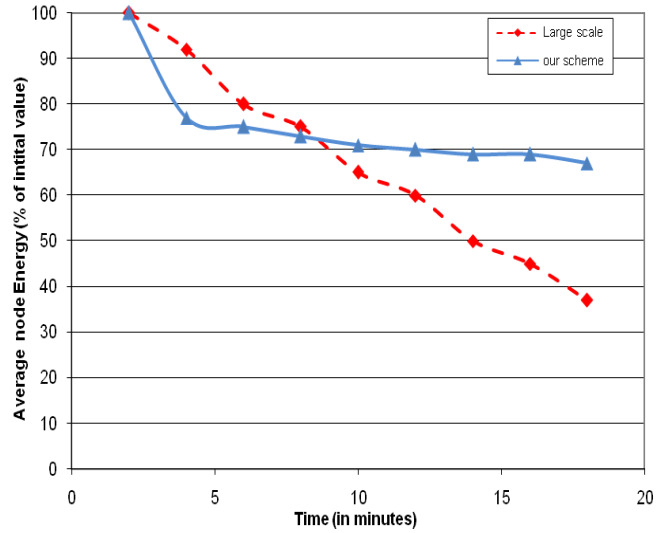


**Figure 10**: Energy loss as we progress in time.

## VII. Conclusion and Future Work

In this paper, we proposed a reactive localization scheme that is both scalable and distributed. The algorithm consists of three consecutive steps and is capable of self-healing. First, we select a special subset of nodes, which we call the anchor nodes, and localize them. The next step is where the reactive concept steps in. When a sensor node detects an event, it reacts by requesting localization information from four anchor nodes. Given this information, the sensor node localizes itself and begins the last step of the Reactive Localization process: delivering the information to the sink. The information delivered to the sink includes the node's location and the description of the event of interest. Many sensor nodes may detect the same event, and this only helps the sink better localize the event. As for self-healing, the algorithm is capable of handling precariously low energy levels and sudden crashes. Analysis and evaluation of our scheme show that it is superior in terms of conserving node energy and hence allowing the system to live longer. It also reduces the communication overhead imposed by other underwater localization algorithms. Localization coverage and errors are comparable to other schemes and at certain node densities slightly better.

As for future work, we plan to find an alternative to localizing the anchor nodes. We are also working on optimizing the number of hops for the sudden-crash self-healing algorithm so that we can minimize the communication overhead.

## References

[1] V. Chandrasekhar, W. K. Seah, Y. S. Choo, and H. V. Ee, "Localization in underwater sensor networks - survey and challenges," in Proceedings of ACM WUWNet'06, Los angeles, CA.

[2] T. C. Austin, R. P. Stokey, and K. M. Sharp. PARADIGM: a buoy-based system for auv navigation and tracking. In IEEE Proceedings of Oceans, 2000.

[3] C. Bechaz and H. Thomas. GIB system: The underwater gps solution. In Proceedings of ECUA, May 2001.

[4] J. E. Garcia. Ad hoc positioning for sensors in underwater acoustic networks. In IEEE Proceedings of Oceans, 2004.

[5] Y. Zhang and L. Cheng. A distributed protocol for multi-hop underwater robot positioning. In IEEE Proceedings of International Conference on Robotics and Biomimetics, 2004.

[6] X. Cheng, H. Shu, Q. Liang, and D. H.-C. Du, "Silent positioning in underwater acoustic sensor networks," in IEEE Transactions on Vehicular Technology, May 2008.

[7] M. Hahn and J. Rice, "Undersea navigation via a distributed acoustic communication network," in Proceedings of Turkish International Conference on Acoustics, 2005.

[8] V. Chandrasekhar and W. K. G. Seah, "Area localization scheme for underwater sensor networks," in Proceedings of the IEEE OCEANS Asia Pacific Conference, 2007.

[9] D. Niculescu and B. Nath, "Ad hoc positioning system (aps)," in GLOBECOM, 2001.

[10] Z. Zhou, J.-H. Cui, and S. Zhou, “Localization for large-scale underwater sensor networks,” in UCONN CSE Technical Report: UbiNet-TR06-04, 2006.

[11] Erol et al. of “AUV-Aided Localization for Underwater Sensor Networks”, International conference on Wireless Algorithims, Systems and applications, 2007, pp: 44-54.

[12] A. K. Othman, A. E. Adams, and C. C. Tsimenidis. Node discovery protocol and localization for distributed underwater acoustic networks. In Proc. of AICT-ICIW ’06, page 93, Washington, DC, USA, 2006.

[13] Z. Zhou, J.-H. Cui, and A. Bagtzoglou, “Scalable Localization with Mobility Prediction for Underwater Sensor Networks,” in INFOCOM 2008. The 27th Conference on Computer Communications. IEEE, April 2008.

[14] W. Cheng, A.Y Teymorian, L. Ma, X. Cheng, X. Lu, Z. Lu, “Underwater Localization in Sparse 3D Acoustic Sensor Networks,” in INFOCOM 2008. The 27th Conference on Computer Communications. IEEE, April 2008

[15] A. Caruso, F. Paparella, L.F.M. Vieira, M. Erol, M. Gerla, "The Meandering Current Mobility Model and its Impact on Underwater Mobile Sensor Networks," INFOCOM 2008. The 27th Conference on Computer Communications. IEEE, April 2008.

[16] J. Albowitz, A. Chen, and L. Zhang. Recursive position estimation in sensor networks. In *Proceedings of IEEE ICNP*, pages 35–41, 2007.