جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

Event localization in underwater wireless sensor networks using Monitoring Courses

Item Type	Conference Paper
Authors	Debont, Matthew John Robert;Jamshaid, Kamran;Shihada, Basem;Ho, Pin-Han
Citation	Debont, M., Jamshaid, K., Shihada, B., & Ho, P.-H. (2012). Event localization in underwater wireless sensor networks using Monitoring Courses. 2012 1st IEEE International Conference on Communications in China (ICCC). doi:10.1109/iccchina.2012.6356987
Eprint version	Post-print
DOI	10.1109/ICCChina.2012.6356987
Publisher	Institute of Electrical and Electronics Engineers (IEEE)
Journal	2012 1st IEEE International Conference on Communications in China (ICCC)
Rights	© 2012 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Download date	2024-04-16 19:12:45
Link to Item	http://hdl.handle.net/10754/362456

Event localization in Underwater Wireless Sensor Systems using Monitoring Courses

Matthew Debont*, Kamran Jamshaid*, Basem Shihada*, Pin-Han Ho†
* MCSE Division, KAUST, Saudi Arabia, {matthew.debont, kamran.jamshaid, basem.shihada}@kaust.edu.sa
† Dept. of Electrical and Computer Engineering, University of Waterloo, Canada, pinhan@bbcr.uwaterloo.ca

***Abstract*—We propose m-courses (Monitoring Courses), a novel solution to localize events in an underwater wireless sensor network. These networks consists of surface gateways and relay nodes. GPS can localize the position of surface gateways which can then distribute their locations through the network using acoustic modems. Relay nodes are deployed to remain static but due to water currents, floating, and the untethered nature of the nodes, they often suffer from frequent drifting which can result in a deployed network suffering link failures. In this work, we developed a novel concept of an underwater alarming system using a cyclic graph model. In the event of link failure, a series of alarm packets are broadcast in the network. These alarms are then captured by the underwater Monitoring Courses (m-courses), which can also be used to assure network connectivity and identify node faults. m-courses also allow the network to localize events and identify network issues at a local level before forwarding results upwards to a Surface Gateway nodes. This reduces communication overhead and allows for distributed management of nodes in a mobile network. Our results show that M-Course routing reduces the number of sends required to report an event to a Surface Gateway by up to 80% when compared to a naïve routing implementation.**

***Index Terms*—Underwater wireless sensor networks, localization, routing**

I. Introduction

Underwater Wireless Sensor Networks (WSNs) can be used in disaster prevention and recovery efforts in deep sea environments deemed unsafe or harsh for human beings. One recent example is the Deepwater Horizon oil spill in the Gulf of Mexico in April 2010. WSNs used in these environments consist of a mix of static and mobile nodes, including nodes tethered to the sea floor at various depths, forming a 3-D network. Most underwater sensor nodes use acoustic telemetry for communication, and thus are limited to low signal propagation speeds and network bandwidth.

In this paper we propose a network monitoring and reliable event localization system for underwater WSNs. While GPS is an accurate and a cost-effective localization solution for terrestrial-based networks, the GPS radio signals do not penetrate large bodies of water [1]. Existing research proposals use Surface Gateways (SGs) with known position (such as a surface buoy or control ship which can use GPS) that then broadcast their location to underwater Relay Nodes (RNs); these nodes then determine their own position using transmission delays and angle of arrival (AoI) of the received signal [2] [3] [4]. Other alternatives include using mobile nodes for localization [2]. Some underwater mobile nodes can periodically become location-aware by surfacing up to get a GPS fix and then rejoin the network to act as a locator beacon for other nodes. Another idea is for nodes to be equipped with a detachable GPS antenna which could float to the surface and then be reeled back once a fix has been made.

A node's location can be used to determine useful information about event localization in a network. For example, if we discover that a drifting node is no longer within communication range, then this information can be sent to a master node so that corrective actions can be taken. Ideally all nodes would know a direct path back to the their local master nodes, but depending on the routing protocol, we may not always know this path beyond a rough direction. Thus, we need to a reliable mechanism of sending information towards a master node. In many cases an event will be detected and pushed towards interested nodes. In this case we rely on beacon nodes to collect these events and make decisions on how to proceed.

Our next goal is to define beacon node placement such that the number of beacons is minimized but also place those beacon nodes in the positions where they will be most useful. This is an NP-Hard problem. Previous solutions [5] consider an acyclic flow network and place beacon nodes so as to increase the probability of an event detection. We aim to develop a similar system, but we do not have the same restrictions and have a different set of problems to deal with.

This paper represents the first phase of a project to develop a fleet of Autonomous Underwater Vehicles (AUVs), with an unprecedented degree of autonomy to carry out a wide range of tasks with little human interaction. These AUVs will communicate over a wireless multi-hop sensor network formed by the fleet itself. We propose a novel underwater node placement scheme and event localization scheme using Monitoring Courses (m-courses). Initially, the nodes are placed in the area in such a way to ensure multiple paths of communication between the surface and a zone-of-interest, while m-courses are generated to update the status of the network and to localize events in the network and report them to relevant controlling nodes.

Our contributions in this paper are as follows:

- A novel algorithm for placing nodes in a 3-D space given a communication range and a zone-of-interest connected to the surface via a series of SGs and RNs
- A routing algorithm that that allows us to group nodes in the network into small clusters and provides a path from any node to at least one SG

- A method of placing beacon nodes in the network as such to ensure that all nodes are covered by a at least one local node, which can be used to determine actions required if needed
- A method to repair the routing system defined above given an non-traversable edge
- A method of broadcasting events from a given node in such a way that beacon or control nodes are aware of the event and can localize it to a given area

The rest of this paper is organized as follows: We introduce m-courses in Sect. II. Sect. III lists the definitions, the network monitoring model, and the problem formulation. Sect. V presents m-courses routing algorithm. Sect. VI describes a novel scheme for beacon node placement and how to interpret data from alarms generated by events in the network. Sect. VII provides a performance analysis of the proposed system. We conclude in Sect. VIII by discussing various open challenges.

II. Monitoring courses (m-courses)

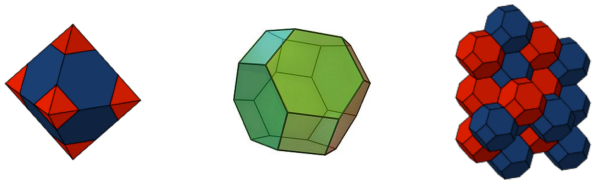
The idea behind m-courses is loosely based upon the concept of Monitoring Trails (m-trails) [6]. A network is divided among a set of cycles, and all nodes and links in the network appear in at least one cycle such that any edge can be identified by the trails which pass through it. When an event occurs, the network is flooded with alarm packets from all trails which identify the event, allowing us to localize the event. This provides fast localization of errors in an all-optical mesh network. However, it has the following drawbacks when applied to underwater WSNs: First, it is costly in terms of power consumption as we need to traverse an edge multiple times to achieve accurate results. Second, the computation time for an m-trail solution is very high. For example, the 21-node ARPA2 topology takes 11234.86 s. to generate an m-trial solution with 9 trails [7]. Such delays are unacceptable for mobile WSNs that may scale to hundreds of network nodes.

Unlike m-trials, m-courses creates a series of cycles in the network, together covering the entire network. It then draws connections across these cycles to create a path from the SG to any other node in the network. These cycles are self-organizing sections of the network with the objective of keeping their own connections and monitoring their members for events.

Another challenge in the formulation of m-courses is the selection of source and destination nodes. One solution is to use static nodes with larger or renewable power supplies (*e.g.*, surface buoys with solar panels) and then simply pick the same node as the source and destination. We could also leverage the fact that the surface buoys may have a more powerful communication network (e.g., radio communications) between each other. This will allow inter-sink communication to transmit results to the original source node.

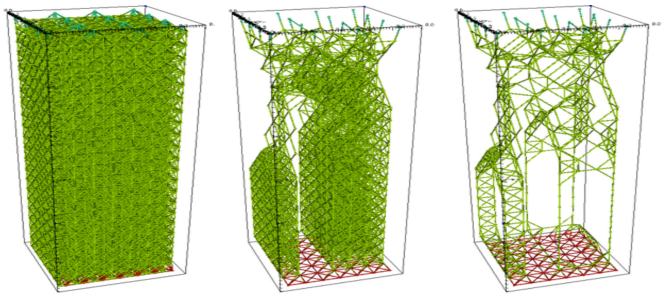
III. Network monitoring model

We have created a network topology generator that places nodes using a Voronoi-Based approach in the space between $Z = 0$ and $Z = ZI$ where ZI is an interface layer between the operational plane and the relay network connecting that



(a) Regular Octahedron (b) Truncated Octahedron (c) Bitruncated Cubic Honeycomb

Fig. 1: Primitives used for node placement in a 3-D Network



(a) Honeycomb (2135 Nodes) (b) Remove unused nodes (832 Nodes) (c) Remove low goodness nodes (235 Nodes)

Fig. 2: Visualizing node placement in a 3-D Network

plane to the surface, which may be the sea floor or simply a zone-of-interest in which we are operating. We have set up the generator to create a range of nodes per local partition, which is simply placed as a square-centered on the SG, which are distributed equally among the surface plane. We can then simply check if each node (V) is in range of any other node to create the edges (E) of the network such that we have a deployed network.

Our approach uses Truncated Octahedrons to form an ideal deployment based on the work shown in [8]. A truncated octahedron itself can be constructed by taking a regular octahedron (shown in Fig. 1a [9]) of edge length $3a$ and removing a square-based pyramid of edge length a from each point (the final result is shown in Fig. 1b [9]). If we take a sphere of radius T_{max} and use this to create a bounding circumsphere to generate a Truncated Octahedron with the node based at the center of this shape, we can tessellate these shapes to form a Bitruncated Cubic Honeycomb (as shown in Fig. 1c [10]) to act as a space-filling polygon for the entire operational area. This gives an optimal placement scheme, but at the cost of a large number of nodes [8]. We can then place a series of SGs on the surface plane by taking a square, which is circumscribed, by a circle based on the maximum range of the acoustic modems.

We then try to find a route from every SG to the interface nodes that form the gateway into the ZI. We then remove nodes that are never touched in this process and then we can try to remove nodes in such a way as to retain connectivity but remove high levels of redundancy. (*e.g.*, nodes with edges greater than a threshold γ.

Fig. 2 shows a network example starting with more than

2000 nodes. It is first reduced by excluding unused nodes and then further reduced by the calculated goodness metric. The result is a reduction to 235 nodes in this example, most of the cuts were made on the basis of nodal degree in this case. This number can be further reduced by making further tests.

IV. Model Definitions

We propose the creation of a wireless acyclic network with two pre-defined points; a SG, which exists at the surface of the zone-of-interest, and a group of Interest Nodes (INs), which exist at the point of interest in that area. We use RNs to enable wireless communications between these points.

A. Definitions

We consider an acyclic graph $G(V, E)$ in which every edge $E(u, v)$ has a non-negative real-valued capacity $c(u, v)$; three sets of vertices, $R = [r_1, r_2, , r_k]$ a set of relay vertices, $D = [d_1, d_2, , d_k]$ a set of source/sink vertices and $I = [i_1, i_2, , i_k]$ a set of IN vertices where $S, D, I \subset V$.

Definition 1: An SG (SG_i) is a surface node with coordinates (x, y, 0). It acts as a gateway between the surface and the acoustic networks. It also offers localization services to the acoustic network. Every SG_i will act as a source node where they inject packets into the network. Our m-course routing algorithm uses them as both source and sink nodes.

Definition 2: An Acoustic Node (AN_i) is an underwater node, *i.e.*, its coordinates are (x, y, z), where z < 0.

Definition 3: A Zone-of-Interest (I) is a subset of edges in graph $G(V, E)$ where current operations are located.

Definition 4: A Relay Node (RN_i) is a node which is an Acoustic Node but not within I. Relay nodes may be declared as beacon nodes in the m-course algorithm.

Definition 5: An Interface Layer (ZI), which acts as an interface between the RN and the Interest Nodes (IN_i) that operate in this area performing a given task.

B. Problem Formulation

The problem we address is how to localize and detect events in a given network of nodes. There are three challenges that initially need to be solved:

1) *SG placement Problem*: Given an operational plane R, calculate the minimum number of SGs required for complete coverage of area via acoustic modems without inter-cell interference. SGs should also be able to communicate via conventional radio modems above the surface.
2) *RN placement Problem*: Given a operational area F, a Zone of Interest I in this area, a threshold value γ and a set of SGs deployed along a plane at the surface R, compute the minimum number of nodes that need to be deployed in F and their deployment locations $p_{x,y,z}$.
3) *Event Localization Problem*: Given a time-varying network F create a tour of all known nodes in F. We can then use this tour to localize various events such as environmental events or link failures.

The given problem considers a partition in a time-varying underwater network. A simpler version of the problem considers an equivalent, static network. We first show that the simplified problem can be described as a Traveling Salesman Problem (TSP), which is a well-known NP-Hard problem.

Theorem 1: The m-course generation problem is NP-Hard

Our problem considers the graph $G(V, E)$ consisting of a set of Vertices $V(x, y, z)$ connected via a set of edges $E(V_i, V_j, c)$ with an associated cost c with a source node S acting as the ingress node. We must ensure that each V is a member of at least one cycle and that every cycle can reach at least one other cycle, in such a way that minimizes the cost of traversing the cycle and each V is touched a minimal number of times. Clearly, this is a variant of TSP, which is a well-known NP-Hard problem. There is no polynomial time algorithm to produce an optimal solution. We instead divide the problem into two non-overlapping sub-problems: First, we present an algorithm to create a set of Cycles (C) called sub-m-courses ensuring full coverage of all $V \in G$. Secondly, we develop a method of linking sub-m-courses into one or more complete cycles called m-courses which both start at end at one or more S nodes. If we assume that both problems 1 and 2 are solved giving us a deployed network $G(V, E)$ we can combine both of these problems into problem 4 as follows:

4) M-Course Generation Problem: Given a network $G(V, E)$ with SG, RN, and a ZI, create a tour or tours that can be used to verify the integrity of $G(V, E)$ and localize events in the network. These tours should be made up of cycles joined together via links to alternate cycles. This can be performed over an entire network or a small partition of the nodes in that network. If we are dealing with a small partition (P) we may have to use nodes, which are external to $P(V_{ext})$, to complete connectivity inside of P.

This algorithm depends on the choice of any E from a given V and to choose we need to determine a 'goodness' value for each E. The basic cost value given for each E can be calculated as a product of the distance between the nodes connected via this edge and SINR. This cost value can then be augmented by conditional options such as;

- Edge leads to a node that does / doesn't exist in local partition
- Edge has been previously visited in the current M-Course Routing run
- Edge leads to the Source node of this particular Sub-M-Course
- Edge leads to a node which is closer to the average XYZ position of the remaining unvisited nodes

We need to determine how each of these attributes contribute to creating a cycle either over a set of / all nodes in P, we can do this be using a series of scalable weights to modify the cost of an edge to create an ideal Sub-M-Course.

Once we have a deployed M-Course solution in a network we need to determine which nodes can be denoted as beacon nodes in this solution, leading us to the next problem;

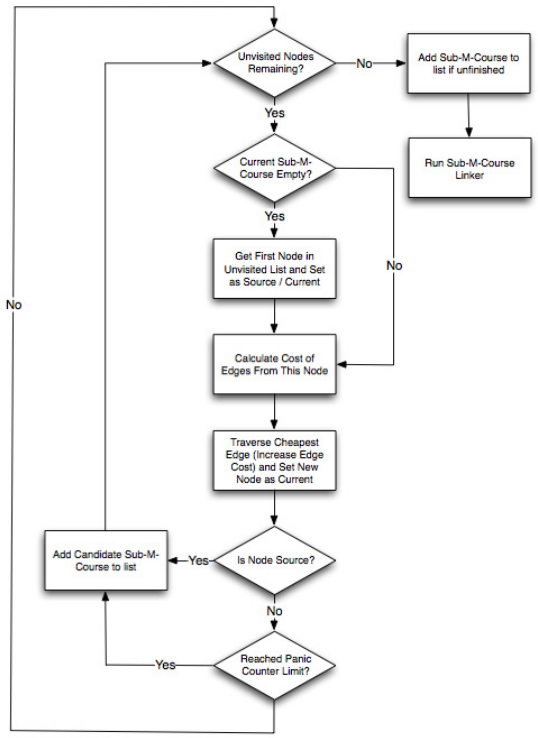


Fig. 3: Basic m-course routing algorithm

5) Given a set of m-courses formed from a set of sub-m-courses SMC over a given time varying network $G(V, E)$ create a set of beacon nodes in the network such that an event can be received and processed by at least one beacon node in each SMC, but also such that a given Degree of Coverage D_c is satisfied allowing an event to be detected and localized accurately.

We can see that this problem is a variant of the problem described in [5]. In our case, we can use the layout of sub-m-courses to help us decide optimal locations as well as the node connectivity.

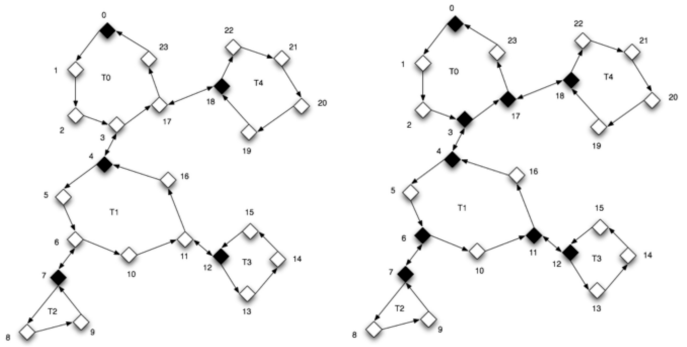
V. M-Course Routing

In this section we describe how to select the source/destination nodes of the m-courses, how to generate sub-m-courses, and finally how to link them together into a single master m-course.

The first problem can be solved by selecting the set of deployed SGs as source/sink nodes. This reduces the problem to simply finding a route from start to finish and then increasing the number of m-courses to provide network-wide coverage.

The next problem is how to create sub-m-courses in the network. In this work we use localized partitioning on the network. This reduces the complexity of this task, helping us create shorter cycles and more efficient m-courses.

A basic algorithm for m-courses routing is shown in Fig. 3. It is loosely based on the Nearest Neighbor algorithm where the cheapest edge is taken from any current edge until we complete the cycle. This may, however, create in-optimal solutions [11]. In such cases we propose using the location



(a) SC providing complete node coverage. Black nodes denote a source for each SC

(b) Candidate Beacon node positions for C shown in (a)

Fig. 4: Naïve beacon placement example

of nodes to steer the sub-m-courses in the direction of other nodes in the population. The sub-m-course generator should run until we visit all nodes in the population, creating a new sub-m-course every time we reach the source node or pass a 'panic' threshold. Once a set of sub-m-courses are created, we can join them into a master m-course that covers all nodes in a local partition. To perform this 'linking' procedure we simply have to follow the sub-m-course containing the SG and check each outgoing edge for nodes which belong to another sub-m-course. Upon discovering an appropriate edge, we join them together and then continue following the path of the sub-m-courses until all sub-m-courses are linked.

A disjoint in connectivity between sub-m-courses can be handled in two ways: (1) Join the two disjoints using links external to the local partition; (2) Make two m-courses and attempt to link the lower m-course with a different SG.

VI. Event Detection and Localization

In this section, we provide a novel scheme for beacon node placement and alarms for event localization using M-Course routing to direct event alarms to an SG.

A. Beacon Placement

We consider a network $G(V, E)$ covered by one or more m-courses (C) made up of one or more sub-m-courses (SC). From the vertex set V, we select a set of Beacon nodes B which act as collection points for Alarm Codes A. Consider the network in Fig. 4a. We create three SCs shown in Fig. 4b (Source nodes for each SC are filled black). These three SCs form one C, which spans all nodes but re-uses four edges in the process as shown in Fig. 4a. This gives us eight candidate beacon node positions (denoted by black nodes in Fig. 4b), which are placed at the intersections between SC to ensure coverage. Ideally we want at least one beacon node in every SC so monitoring can be dealt with locally. In this particular example, however, this leads to beacon nodes being close together creating pairs of tightly coupled nodes at the intersections between SC.

In order to find the optimal beacon node positions we adapt an idea from [5]: using an initial candidate pool of beacon

nodes (*e.g.*, SG nodes), we use a potential value (ϕ) which is based upon the parent nodes of the current vertex such that for two vertices v_i and v_j, v_j is the parent of v_i, $v_i\dot{\phi} > v_j\dot{\phi}$. We use a threshold value (τ) so that once a node's ϕ falls below τ the node at that vertex is turned into a beacon node reducing the τ value of all subsequent nodes. The value of τ cannot be calculated in the same fashion as in [5] since packets are always guaranteed to flow down a pre-described edge if that edge exists in the network. Using differing values of τ we can attempt to regulate the number of beacon nodes in the network. We can use this concept in tandem with the naïve placement model to create a working model. We place a beacon node pre-emptively at the starting node of each Sub-M-Course and then run the algorithm, adding extra beacon nodes into the network when needed. This system allows us to distribute beacon nodes into the network acting as buffers for events and placing these beacon nodes in useful positions when there is not enough coverage in the local area.

B. Alarms

We use a system of alarms to inform the nodes of a sub-m-course of an event. When a node detects an event, it forwards the alarm along the local sub-m-course until it reaches a beacon node in that sub-m-course. The beacon node can then forward the alarm upwards towards the SG.

1) Environmental Alarms: If a node detects an environmental variable of interest, it generates an alarm, transmitting information identifying the ID of the sub-m-course, the type of alarm, as well as the actual alarm data. This information is passed along to the next node in the sub-m-course until it reaches a beacon node responsible for transmitting it upwards in the dependency tree towards the SG.

A beacon node may buffer the received alarms in an attempt to aggregate similar alarms as detected by other nodes in its sub-m-course. It can then then send this batch of alarms up the network, following the m-course up to the SG node.

2) Network Alarms: Network alarms are triggered in response to network issues. We describe two use cases: (1) when a node in a sub-m-course can no longer communicate with the next hop; (2) when a node cannot find the next Sub-M-Course. *Node Specific Alarms*: When a node in a network drifts away, its immediate neighbors can transmit an alarm packet containing its sub-m-course information. Nodes belong to other sub-m-courses can safely discard this message. These alarms will be collected at beacon nodes in the network so that the node management protocol can be used to resolve issues. It is preferred to handle such disconnections locally so as to avoid informing the SG or other sub-m-courses of the resulting actions being taken to keep connectivity.

Sub-M-Course Specific Alarms: Links between sub-m-courses act as local gateways links which if broken lead to network partitioning. If this link is broken, we send an alarm code for a missing SC. If a node on the target SC receives this alarm, it will try and complete its pre-defined route, pre-emptively promoting itself to a beacon node and acting as a new ingress node for this Sub-M-Course.

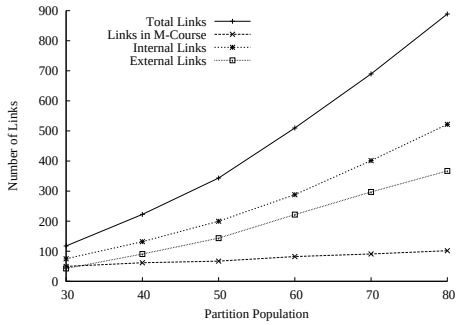


Fig. 5: Total links vs. links in generated m-courses

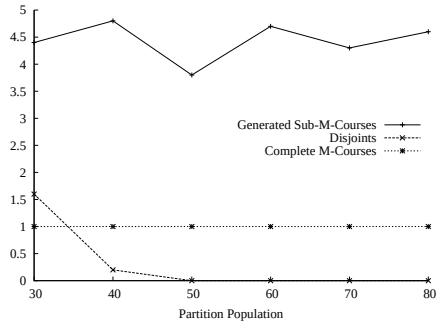


Fig. 6: m-course statistics with increasing node density

C. Multiple Events

These mechanisms can be extended to support multiple events in a network. For environmental alarms, we would collect the alarms at local beacon nodes where they will be handled as identified above. Handling multiple network alarms only becomes an issue if there are multiple alarms in the same sub-m-course since alarms in separate sub-m-courses should be solved internally by the beacon nodes responsible for those sub-m-courses. Multiple alarms inside a single sub-m-course can still be recovered as long as a viable set of edges remain within the sub-m-course.

VII. Performance evaluation

We have evaluated the performance of various algorithms proposed in this work using simulations. We have tested them over 100 random networks with differing node densities. The m-course output was generated and visualized through the VisIt visualizer [12] giving us a blow-by-blow representation of how any particular M-Course is generated from individual sub-m-courses to linking and then to the final m-course itself.

A. M-Course Generation Results

We consider various randomly generated networks in our algorithms with increasing node density inside of a local partition to try and increase the number of existing paths between nodes. We started with a low node density of 10 RNs per partition; however this created networks which were unviable. We found that random generation only seems to generate viable networks when we have around 40 nodes per partition with the given placement area and maximum range values. Our results are summarized in Figs. 5 and 6. We can

see that the number of links in a given m-course increases in an almost linear fashion with the density of nodes in local partition. We can also see that we are generating on average four to five sub-m-courses per local partition. The algorithm can be tuned to give more or fewer sub-m-courses as required.

B. Event Localization Performance Results

We simulated an event by creating a new node at a random location in a network. Once the event is detected, alarms are generated. We compare the number of transmissions needed in our proposed system with a simple and naïve system which simply rebroadcasts received alarms towards an SG where the event can be recorded. We expect our approach to require fewer broadcasts as we only consider one path. While this path may not be optimal, but this method allows us to collect alarms at beacon nodes in the network before forwarding them, thus further reducing the number of broadcasts.

Assuming a perfect channel in which no retransmissions are required, completing an m-course requires at most *n* broadcasts, where *n* is the number of nodes in the network. This cannot be reduced in this scheme however as each node must send at least once. However, once an event is detected and alarms are broadcast, a naïve system would at most require *n* transmissions to ensure delivery. However, using our proposed dependency tree we can reduce the number of send operations by only sending the alarm to sub-m-courses above us in the dependency tree and forwarding the alarm along the prescribed m-course towards the SG. This requires at most $SMC_V + SG_{Path}$ sends (where SMC_V refers to the set nodes in this sub-m-course and SG_{Path} is the number of nodes in the path leading to SG.

We experiment with a large network and generate seven events at various depths such that each event will hit at least one sub-m-course. Our results are summarized in Figs. 7 and 8. m-courses routing may increase the time taken for an event to reach a SG. This is in part due to the buffering effect where we wait for all alarms to arrive before sending the alarms forward. However, m-courses also significantly reduces the number of broadcasts required for any particular event to reach an SG. Further, we note that the naïve solution does not cope well with multiple alarms, especially shown in the event 7. Thus m-course routing is independent of the number of alarms and reduces the number of send operations required for single and multiple events.

C. Network Alarms

If we take the example M-Course described in Fig. **??**, we can simulate a network error by removing one of the nodes in the network and then using the techniques described in the previous chapter to attempt to fix the issue.

We can see that this approach works well in simple examples and in this case we do not require a retransmission from the Beacon node of the alarm to ensure coverage as an alarm has successfully traversed part of the Sub-M-Course and arrived back at the Beacon node. In this case we would send the new path along the Sub-M-Course to update the routing

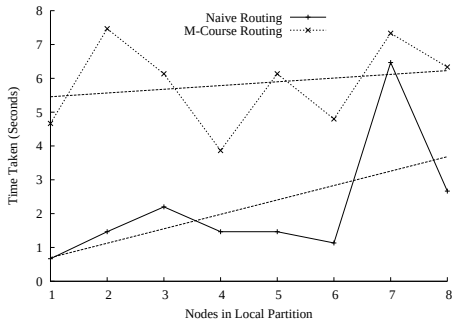


Fig. 7: Time taken by event alarms to reach SG

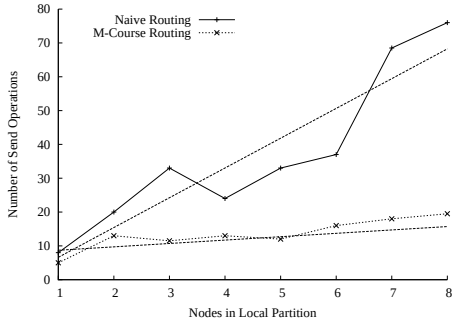
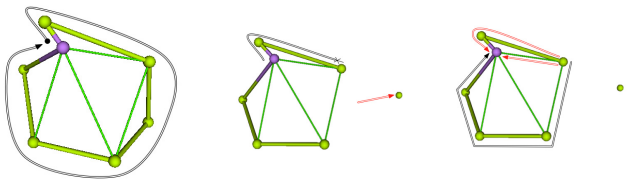


Fig. 8: Number of sends for event alarms to reach SG

mechanism but would also then send an update upwards to the SG to inform the control nodes of the error. The cost of recovering from this event is directly linked to the number of edges in the Sub-M-Course and interconnecting edges in that Sub-M-Course, in this case the maximum cost would have been if the second node of the Sub-M-Course where to drift away, causing the beacon node to have to send out an alarm after a time out as it would not be possible for the alarm to reach the beacon along the remaining Sub-M-Course. However, the damage would be repaired. This cost is greatly dependent on the nature of the event itself. We may still have to broadcast the alarm along all edges if an alarm is received for an edge logically after the current node (i.e., implying that a link further along in the Sub-M-Course is non-traversable). This effect requires us to send the alarm back along the previously proven path. In other words, at most we will have $N + UN$ send operations (where N is the number

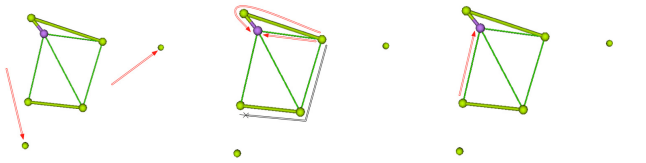


(a) Normal packet routing in case of no errors

(b) One node moves completely out of communication range

(c) Generated Alarms, in this case we have three paths generated, one following the Sub-M-Course after skipping missing node

Fig. 9: Network Alarms across a Sub-M-Course



(a) Second node drifts out of position (b) Initial Alarm path is also disrupted (c) Send out a new alarm based on the new missing edges

Fig. 10: Network Alarms across a Sub-M-Course

of nodes in the Sub-M-Course and U is the number of links which are not used in the Sub-M-Course and connect two nodes which are members of the Sub-M-Course), we require UN as at most we would need to send the data along the whole Sub-M-Course back to the beacon node. From this collected data we would need a further N communications to relay the new Sub-M-Course path to the nodes. Therefore, to recover from any one network alarm we need at most $2N + UN$ send operations. In reality this is very unlikely to happen, recall the example shown in Fig. 9. It requires 13 send operations to fully recover (6 to distribute the new Sub-M-Course and 7 alarm related send operations).

D. Multiple Network Alarms

We expand on the example shown in Fig. 9 to include a second event as shown in Fig. 10. In this case we would store the initial alarm path at the source of the second alarm and proceed as before. When the beacon node receives the new path from the node, it would request the remainder of the first alarm to complete its picture of the entire network.

The cost of this operation is also greatly dependent on the remaining Sub-M-Course as before. It may require an even greater number of communications depending on the topology. If we use this idea recursively, such data should always make it back to the beacon node for processing. The cost will likely increase rapidly, and we will need to devise a better solution for tracking multiple network events in the same Sub-M-Course. In the example shown in Fig. 10 we would require 12 send operations to recover fully from these errors (5 to distribute the new Sub-M-Course and 6 as a result of alarm related sends).

VIII. Conclusions and Future Work

We have proposed a novel method of creating Monitoring Courses (m-courses) over a set of underwater WSNs. Our approach is tolerant to disjoints of connectivity inside a network. These m-courses can be described as a type of acyclic flow network with cycles acting as a routing tree allowing events to be collected and dealt with at a local level. It can discover edges that are no longer reachable due to node movement or malfunction. We have also described a method of broadcasting alarms over the network in such a way that nodes can be rediscovered if still in the network.

For future work, we plan to improve the sub-m-course generation algorithm through the use of other cycle generating techniques, to create smaller sub-m-courses with increased interconnectivity, allowing for more redundant paths between sub-m-courses. We found that our proposed solution can have a subset of RNs which are primarily used for communication and thus suffer a much greater level of power drain. We can alleviate this problem by reversing the flow of the Sub-M-Course to push messages into lesser used portion of the network. We could also attempt to leverage the mobility of the nodes to push nodes with a greater battery life towards the upper layers of the network. We plan to address these issues in follow-on work.

References

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

[2] M. Erol, L. Vieira, and M. Gerla, "AUV-Aided Localization for Underwater Sensor Networks," *Int Conf on Wireless Algorithms Systems and Applications WASA*, pp. 44–54, 2007.

[3] Z. Zhou, J. H. Cui, and A. Bagtzoglou, "Scalable Localization with Mobility Prediction for Underwater Sensor Networks," in *Proceedings of the IEEE Infocom*, July 2008, pp. 2198–2206.

[4] Z. Zhou, J.-H. Cui, and S. Zhou, "Efficient localization for large-scale underwater sensor networks," *Ad Hoc Networks*, vol. 8, no. 3, pp. 267–279, 2010.

[5] M. Suresh, R. Stoleru, R. Denton, E. Zechman, and B. Shihada, "Towards Optimal Event Localization in Time-Varying Acyclic Flow Networks," in *Proceedings of the ICDCN*, 2012.

[6] J. Tapolcai, B. Wu, P.-H. Ho, and L. Ronyai, "A Novel Approach for Failure Localization in All-Optical Mesh Networks," *IEEE/ACM Transactions on Networking*, vol. 19, no. 1, pp. 275–285, 2010.

[7] B. Wi, P.-H. Ho, and K. L. Yeung, "Monitoring Trail: A New Paradigm for Fast Link Failure Localization in WDM Mesh Networks," in *Proceedings of the IEEE Globecom*, 2008.

[8] S. Alam and Z. J. Haas, "Coverage and connectivity in three-dimensional underwater sensor networks," *Wireless Communications and Mobile Computing*, vol. 8, no. 8, pp. 995–1009, 2008.

[9] Wikipedia, "Bitruncated Cubic Honeycomb," 2011. [Online]. Available: http://en.wikipedia.org/wiki/Bitruncated_cubic_honeycomb

[10] ——, "Truncated Octahedron," 2011. [Online]. Available: http://en.wikipedia.org/wiki/Truncated_octahedron

[11] L. Tassiulas, "Worst case length of nearest neighbor tours for the Euclidean traveling salesman problem," *SIAM Journal on Discrete Mathematics*, vol. 10, pp. 171–179, 1997.

[12] Lawrence Livermore National Laboratory, "Visit Visualization Tool," 2011. [Online]. Available: https://wci.llnl.gov/codes/visit/