

Void Avoidance in Mobile Underwater Sensor Networks

Peng Xie, Zhong Zhou, Zheng Peng, Jun-Hong Cui and Zhijie Shi

{xp,zhongzhou,zhengpeng,jcui,zshi}@cse.uconn.edu

Computer Science & Engineering Department, University of Connecticut, Storrs, CT 06269

Abstract

Mobile underwater sensor networks are featured with three-dimensional topology, high node mobility and long propagation delay. Geographic routing has been shown to be very suitable for such networks. However, routing voids pose great challenges on the greedy policy used in most geographic routing protocols. This is especially true for the underwater sensor networks because of node mobility and three-dimensional topology. In this paper, we propose a void avoidance protocol, called Vector-Based Void Avoidance (VBVA), to address the void problem in mobile underwater sensor networks. VBVA adopts two mechanisms, vector-shift and back-pressure, to handle voids. Vector-shift mechanism is used to route data packets along the boundary of the void. Back-pressure mechanism routes data packets backward to bypass a concave void. VBVA handles the void problem on demand and thus does not need to know network topology and void information in advance. Therefore, it is very robust to cope with mobile voids in mobile networks. To the best of our knowledge, VBVA is the first void avoidance protocol to address the three-dimensional and mobile void problems in underwater sensor networks. Our simulation results show that VBVA works effectively and efficiently in mobile underwater sensor networks.

I. INTRODUCTION

Underwater sensor networks have received growing interests recently [2], [5], [9], [18], [19]. In some application scenarios such as estuary monitoring and submarine detection, mobile underwater sensor networks are urgently demanded [5], [12]. In these scenarios, mobile underwater sensor networks empower people to monitor or detect phenomena more accurately and timely in extended areas. In such networks, the sensor nodes are deployed in a three-dimensional space. The majority of sensor nodes, except for some nodes equipped with surface-level buoys, are mobile due to water current and other underwater activities. From empirical observations, underwater objects typically move at the speed of 2-3 knots (or 3-6 kilometers per hour). Efficient routing faces great challenges in such three-dimensional mobile networks.

In [16], [22], it is shown that geographic routing protocols such as VBF [22] are suitable for mobile underwater networks. Geographic routing protocols rely on the geographical information of the nodes to determine the next data-forwarding node. These protocols have explored various greedy policies for selecting the next forwarding node. However, most of the greedy policies, which try to select one or more neighbors that are the nearest to the destination, may not be effective or efficient in certain network scenarios. For example, in sparse networks, a node probably cannot forward its data packets based on

its greedy policy if none of its neighbors is closer to the destination. This phenomena is referred to as a routing void in [6], [15]. How to cope with routing voids in geographic routing protocols is quite challenging.

The characteristics of underwater sensor networks make the problem even more difficult. First of all, a void in underwater sensor networks is three-dimensional. Second, the mobility of most underwater nodes makes the void mobile. A mobile void can also result from the surrounding environment. For example, when a ship navigates over the underwater sensor network, it blocks communications in the nearby area and thus generates a void that moves along with the ship. The characteristics of underwater sensor networks make it more difficult to cope with three-dimensional and mobile voids in such networks.

In this paper, we propose a new void avoidance routing protocol, vector-based void avoidance (VBVA) routing protocol, to address the void-avoidance routing in mobile underwater sensor networks. VBVA is a vector-based approach. Initially, the forwarding path of a data packet is represented by a vector from the source to the destination. If there is no presence of voids in the forwarding path, VBVA is essentially the same as vector-based forwarding (VBF) [22]. When there is a void in the forwarding path, VBVA adopts two methods, vector-shift and back-pressure, to handle the void. In the vector-shift method, VBVA attempts to route the packet along the boundary of the void by shifting the forwarding vector of the data packet. If the void is convex, vector-shift method can successfully route the packet around the void and deliver it to the destination. However, if the void is concave, the vector-shift method may fail. In this case, VBVA resorts to back-pressure method to route the data packet back to some nodes suitable to do vector-shift. We prove that if the network is connected and stable in the time period of delivering the data packet, VBVA can always find an available path. Since VBVA avoids voids on demand, i.e., VBVA does not rely on the topology information of the network, it can handle the mobile network and mobile voids efficiently and effectively. To the best of our knowledge, VBVA is the first protocol to address the three-dimensional and mobile void problem in underwater sensor networks.

The rest of the paper is organized as follows. In Section II, we review existing work on the routing protocol for underwater sensor networks and the void avoidance algorithms. Then, we present VBVA in detail in section III. We evaluate the performance of VBVA in different network settings in Section V. In section VI, we conclude our work and point out the future work.

II. RELATED WORK

In this section, we will first review routing protocols for underwater sensor networks. After that, we will review existing void avoidance algorithms and show their differences from our work.

A. Routing protocols in underwater sensor networks

A lot of research has been done in the last few years on the routing protocols for underwater networks. The challenges and the state of arts for the routing protocols in underwater networks have been discussed in details in [9], [5]. A pioneering work is done in [21] on the routing protocol for underwater networks. In this work, a central master node is used to probe the network topology and do the route establishment. The authors of [17] propose a centralized routing algorithms for delay sensitive application and a distributed

routing algorithm for delay-insensitive applications in three-dimensional underwater networks. In [1], the authors propose a novel method to improve the efficiency of the flood-based routing protocol in underwater sensor networks. Focus beam routing appears in [10], which dynamically establish a route as the data packet traverses the network towards its final destinations. An adaptive routing protocol for underwater Delay Tolerant Networks(DTN) has been proposed in [8], which divides the network into multiple layers and every node adaptively find its routes to the upper layer according to its past memory. Vector-Based-Forwarding (VBF)protocol appears in [22], which takes advantages of the location information to form one or multiple routing pipes from the source to the destination. Multiple routes might be used simultaneously in VBF to improve the reliability.

Different from all of the above work, our VBVA protocol is a geographic routing protocol with void avoidance capacity. To the best of knowledge, VBVA is the first geographical routing protocol which can deals with voids efficiently in underwater sensor networks.

B. Void avoidance algorithm

There are numerous work proposed to handle the void problem in radio sensor networks. In [11], [23], a graph traversal algorithm *right-hand rule* is proposed to bypass the void in the network. A node bypasses a void by forwarding the data to the node first traversed by the arriving edge of the packet counterclockwise. However, this algorithm requires that the underlying graph is planar. A flooding algorithm appears in [20]. Here, when a node cannot forward a packet further, it floods the packet one-hop, and then its the neighbors will forward the packet in a greedy way. Here, a node needs to know its neighbors' location information and only the node with best position is selected as the next hop. Distance upgrading algorithm (DUA) is proposed in [4]. In this algorithm, each node records its virtual distance to the destination. A packet is forwarded from the nodes with larger virtual distances to the nodes with lower virtual distances. When a node finds out that it is a dead end, it increases its virtual distance to prevent others from sending more packets to it.

However, all of the existing void avoidance routing protocols address stationary and two-dimensional wireless networks. They are not suitable for three-dimensional mobile underwater sensor networks, which are the target of the VBVA protocol.

III. VECTOR-BASED VOID AVOIDANCE PROTOCOL

In this section, we describe our vector-based void avoidance algorithm (VBVA). VBVA extends the vector-based forwarding protocol(VBF). When there is no void, VBVA works in the same way as VBF. If a void is detected, VBVA resorts to its vector-shift mechanism which tries to shift its forwarding vector to bypass the void. If the vector-shift mechanism does not work, VBVA will use its back-pressure mechanism to retreat the data packet back to some nodes that can use vector-shift mechanism to bypass the void. In the next, we will first give a brief introduction to VBF. Then, we will discuss how to detect a void in VBVA and present its vector-shift and back-pressure mechanism. At the end, we will describe the VBVA protocol in detail.

A. Vector-Based Forwarding (VBF)

VBF is a geographic routing protocol [22]. Each node in the network knows its position, which can be obtained by localization algorithms [3], [24], [7], [13], [26], [25]. If no localization service is available, the sensor nodes can still estimate its relative position to the forwarding node by measuring its distance to the forwarder and the angle of arrival (AOA) if it is equipped with some hardware device. This assumption is justified by the fact that acoustic directional antennae are much smaller than RF directional antennae due to the extremely small wavelength of sound. Moreover, underwater sensor nodes are usually larger than land-based sensors and thus have room for such devices.

In VBF, the forwarding path follows a vector from the source to the target, which is called forwarding vector. The position information of the source, target, and forwarder is carried in the header of the data packet. When a node receives a packet, it calculates its distance to the forwarding vector. If the distance is less than a pre-defined threshold, called radius, this node is qualified to forward the packet. In VBF, the forwarding path is virtually a pipe from the source to the target, called forwarding pipe.

In order to suppress the duplication, VBF adopts a self-adaptive algorithm. Each qualified node holds the packet for some time period determined by its position. The node that is the closest to the forwarding vector and to the destination forwards the packet first. Upon overhearing the transmission of the packet, other qualified nodes determine whether to forward the same packet by weighing their benefit to do so.

VBF is very robust against mobile networks, error-prone channel and vulnerable sensor nodes. However, VBF cannot route packets around a void. The variant of VBF, HH-VBF improves the robustness of VBF against sparse networks [16]. In HH-VBF, each forwarding node adopts the forwarding vector starting from itself to the target. If a node is inside the forwarding pipe from the forwarding node to the target and closer to the target, the node is qualified to be the next hop. Although HH-VBF is more robust than VBF [16], it still suffers from the void problem due to its greedy policy, especially when the source is located in a concave void.

VBVA extends VBF with the capability to handle voids. It has the same assumptions as VBF, i.e., a node can overhear the transmission of its neighbors. Because of broadcasting nature of underwater acoustic channels, this can be easily satisfied. Similar to VBF, the forwarding path of a packet is also represented as a vector and carried in the packet in VBVA. If there is no presence of void in the forwarding path, VBVA behaves the same as VBF. However, VBVA significantly differs from VBF in that VBVA can detect the presence of the void in the forwarding path and bypass the void. With its void avoidance mechanism, VBVA can potentially find multiple forwarding vectors for a data packet, thus significantly improving the robustness of the network.

B. Void Detection

In VBVA, a node detects the presence of a void by overhearing the transmission of the packet by its neighboring nodes. In VBVA, the information about the forwarding vector of a packet is carried in the packet. When a node overhears the transmission of a data packet, the node records the position information of the forwarding nodes. We denote the start point and the end point of the forwarding vector by S and

T respectively. For any node N , we define the *advance* of node N on the forwarding vector of the packet is the projection of the vector \overrightarrow{SN} on the forwarding vector \overrightarrow{ST} . We call a node a *void node* if all the advances of its neighbors on the forwarding vector carried in a packet are smaller than its own advance. An example is shown in Fig. 1, the forwarding vector of a packet is \overrightarrow{ST} , the advances of nodes B, C and F on the forwarding vector are denoted as A_B , A_C and A_F , respectively. As shown in Fig 1, all the neighbors of node F have smaller advances than F on the forwarding vector \overrightarrow{ST} . Thus node F is a void node. In VBVA, if a node finds out that it has the largest advance on the current forwarding vector among all the neighboring nodes within the forwarding pipe, the node concludes that it is a void node and has detected a void on the current forwarding vector.

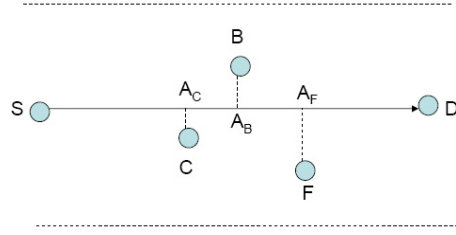


Fig. 1. An example of void node

It is clear that a void node is essentially an edge node that neighbors a void in the forwarding direction of a packet and cannot forward the packet any further in the direction to the target along the current forwarding vector.

Unlike VBF where the forwarding vector of a packet is defined to be the vector from the source to the target and is kept unchanged during the packet delivery. In VBVA, once a forwarding node detects a void, the node tries to bypass the void by changing the forwarding vector of the packet to find alternative routes. Two mechanisms, *vector-shift* and *back-pressure*, are used in VBVA for bypassing the void. We describe the two mechanisms in detail next.

C. Vector-shift Mechanism

In VBVA, when a node determines that it is a void node for a packet, it will try to bypass the void by shifting the forwarding vector of the packet first. To do the vector shifting, the node broadcasts a vector-shift packet to all its neighbors. Upon receiving this control packet, all the nodes outside the current forwarding pipe will try to forward the corresponding data packet following a new forwarding vector from themselves to the target. This process is called *vector-shift* and we say the void node *shifts* the forwarding vector.

As shown in Fig. 2, the dashed area is the void area. Node S is the sender and node T is the target node. At the beginning, node S forwards the packet along the forwarding vector \overrightarrow{ST} , and then it keeps listening the channel for some time period. Since the neighboring node D and A of node S are not within the forwarding pipe, they will not forward this packet. Thus node S cannot overhear any transmission of the same packet and concludes that it comes across a void. It then broadcasts a vector-shift control packet, asking its neighbors to change the current forwarding vector to \overrightarrow{DT} and \overrightarrow{AT} as shown in Fig. 2.

Nodes D and A repeat the same process. The arrowed lines in Fig. 2 are the forwarding vectors used by the forwarding nodes. From this figure, we can see that if the void area is convex, it can be bypassed by the vector-shift mechanism.

After shifting the forwarding vector of a packet, a node keeps listening the channel to check if there is a neighboring node forwarding the packet with the new forwarding vector. If the node does not hear the packet being forwarded even if it shifts the current forwarding vector, the node is defined as an *end node*. For an end node, the vector-shift mechanism cannot find an alternative routing path and we have to use a new back-pressure mechanism.

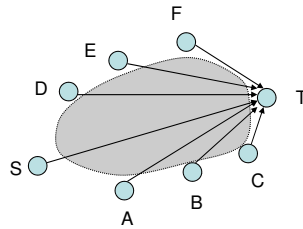


Fig. 2. An example of vector-shift mechanism

D. Back-pressure Mechanism

When a node finds out that it is an end node, it broadcasts a control packet, called **BP**(Back Pressure) packet. Upon receiving a **BP** packet, every neighboring node tries to shift the forwarding vector of the corresponding packet if it has never shifted the forwarding vector of this packet before. Otherwise, the node broadcasts the **BP** packet again. We call the process of repetitively broadcasting the **BP** packet the *back-pressure process*. The **BP** packet will be routed back in the direction moving away from the target until it reaches a node which can do vector shifting to forward the packet toward the target.

Fig. 3 shows an example for the back-pressure process. Here, the dashed area is a concave void. The node S is the sender and node T is the target. When node S forwards the packet with forwarding vector \vec{ST} to node C, since node C cannot forward the packet along the vector \vec{ST} any more, it will first use vector-shift mechanism to find alternative routes for the data packet. Since node C is an end node, it cannot overhear the transmission of the packet. Node C then broadcasts a **BP**(Back Pressure) packet. Upon receiving the **BP** packet, node B first tries to shift the forwarding vector but fails to find routes for the data packet. Then node B broadcasts **BP** packet to node A and so on. Finally, a **BP** packet is routed from node A to the source S. Node S then shifts the forwarding vector to \vec{HT} and \vec{DT} . The data packet is then forwarded to the target by the vector-shift method from nodes H and D as shown in Fig. 3. It is clear from Fig. 3 that our back-pressure mechanism can handle the end-node problem and the concave void.

the original data. For example, node A in Fig. 3, after it receives the **BP** packet from node B , it will send a **VSD** packet, trying to shift the forwarding vector first to bypass the void. Since at this time, the neighbors of node A possibly do not keep a copy of the corresponding data packet, node A needs to provide the original data packet in its **VSD** packet. To summarize, a node sends **VSD** for vector shifting if it has received the **BP** packet.

BP, **EPN** and **EPND** control packets are used in the back-pressure mechanism. In the back-pressure process, if the direction from a node to the target leads to a dead end, the data packet has to travel back, in the direction away from the target. In each step of the retreat, a node who has received a **BP** packet will try to use vector-shift mechanism to forward the data packet if it has not done that before. If this node can successfully find a next-hop node through the vector shifting, it is called a *first forwarding node* since it starts a new forwarding path towards the target after a back pressure process. For example, in Fig. 3, when node B receives a **BP** from node C , it will try to forward the packet through vector shifting. However, it fails to find any next-hop node. Thus, node B does not become a first forwarding node and relay the **BP** packets further backward to node A . When node S receives a **BP** from node A , it can successfully find the next-hop nodes H and D through its vector shifting. In this case, node S becomes a First Forwarding Node.

Every time when a back-pressure process traces back to a first forwarding node, if this first forwarding node still cannot find a feasible next-hop node for the packet by itself, it will broadcast a **EPN** or **EPND** to ask for help from all its neighboring nodes. Thus, **EPN** or **EPND** control packets are only sent out by the first forwarding node in the back-pressure mechanism. And nodes located between the first forwarding node and the end node use **BP** in the back-pressure mechanism.

The transmission status of a data packet on a node in VBVA can be FORWARDED, CENTER-FORWARDED, FLOODED, SUPPRESSED or TERMINATED. When a node forwards a packet normally, the transmission status of the packet on this node is FORWARDED. If a node sends an **EPN** or **EPND** packet, the transmission status of the corresponding packet is set to CENTER-FORWARDED (Only the first forwarding node possibly has the CENTER-FORWARDED transmission status). If a node shifts the forwarding vector of a packet, the transmission status of the packet in this node is FLOODED, which means that this node has already flooded the packet along its forwarding vector to the target. If a node drops a packet due to the duplication suppression, this node marks the transmission status of the packet as SUPPRESSED. If a node broadcasts a backward pressure control packet such as **BP**, the transmission status of the corresponding data packet is set to TERMINATED, which means that this node will not process any packets related to this data packet any more.

F. Protocol Process

In this section, we will present our VBVA protocol in detail.

When a source node gets a data packet, it will do as follows.

- 1) The source first broadcasts the data packet, and sets a timer which will time out after the void-avoidance delay.

- 2) The source listens to the channel and records the position and ID of the forwarding nodes of this packet in its next-hop-status table.
- 3) After the timer expires, the source determines if it is a void node by checking its next-hop-status table. If it is, the source broadcasts an **EPN** to ask for help from its neighbors and marks the transmission status of the packet as **TERMINATED**. Otherwise, the source sets the transmission status of the packet as **CENTER-FORWARDED** in its transmission-status table.

Upon receiving a data packet, the node checks whether it has received this packet before, i.e., whether there are entries for this packet in the the next-hop-status table and transmission-status table. If there are, this node simply ignores the packet. Otherwise, the node treats the data packet as a new packet as follows.

- 1) The node first copies the packet in its buffer and keeps it for time period of void-avoidance delay. This node then calculates its distance to the forwarding vector of the packet.
- 2) If the node is inside the current vector forwarding pipe, the node runs the same self-adaptive algorithm as in VBF [22] to determine whether to forward the data packet. If the node decides to drop the packet, marks the transmission status of the the packet as **SUPPRESSED**.
- 3) If the node forwards the packet, then marks the transmission status of the data packet as **FORWARDED**. It then sets a timer which will expire after the void-avoidance delay and listens to the channel.
- 4) Before the timer expires, if the nodes overhears the transmission of the same data packet, this node records the position information and id of the forwarding nodes in its next-hop-status table.
- 5) After the timer expires, the node checks its next-hop-status table to determine if it is a void node. If it is not, this node deletes the data packet from its buffer. Otherwise, The node broadcasts a **VS**, set the transmission status of this packet as **FLOODED**, sets a new timer which will expire after the void-avoidance delay and listens on the channel.
- 6) After the timer expires, this node checks if its vector shifting mechanism works successfully, i.e., it checks if it overhears the transmission of the data packet whose forwarding vector is from itself to the target. If it does not, it then broadcasts a **BP** packet and set the transmission status of this data packet as **TERMINATED**.

It is clear that if a node is a void node, it will first broadcast a **VS** packet to inform its neighboring nodes to do vector shifting. Upon receiving a **VS** packet, the neighboring node will do the following operations.

- 1) The node checks if there is the corresponding data packet in its buffer. If the node cannot find the corresponding data packet, the node ignores the **VS** packet. Otherwise, it goes to Step 2).
- 2) The node checks if it is outside of the current forwarding pipe. If it is not, the node drops the packet. Otherwise, it goes to Step 3).
- 3) The node does vector shifting to change the forwarding vector of the data packet to the direction from itself to the target, records the transmission status of the packet as **FORWARD** and forwards the packet.

VBVA processes **VSD** in the same way as **VS** except that the receiving node does not need to check

its buffer for the data packet. The receiving node can extract data packet directly from **VSD**.

If vector-shift mechanism does not work, the void node will send out a **BP** packet to initiate a back-pressure process. Upon receiving a **BP** packet, a node should first checks its packet-status table.

- 1) If the transmission status of the packet is **FLOODED** and this node is not a first forwarding node, this node generates and broadcasts a **BP** packet and marks the transmission status of the data packet as **TERMINATED**.
- 2) If the transmission status of the packet is **FLOODED** and this node is a first forwarding node, this node generates and broadcasts a **EPND** packet and marks the transmission status of the data packet as **CENTER-FORWARDED**.
- 3) If the transmission status of the packet is **TERMINATED**, the node ignores the packet.
- 4) If the transmission status of the data packet is **CENTER-FORWARDED** or **FORWARDED**, the node first set the item in its next-hop-status table which corresponding to the source of the **BP** packet as **DEAD**.
- 5) If all the items in its next hop are marked as **DEAD**, this node performs the following steps.
 - a) If the transmission status of the packet is **FORWARDED**, this node generates and broadcasts a **VSD** packet. Change the transmission status of the packet to **FLOODED**. If at least one next-hop node is found by its vector shifting mechanism, this node becomes a new first forwarding node.
 - b) If the transmission status of the packet is **CENTER-FORWARDED**. this node generates and broadcasts an **BP** packet

As shown in the above, when the back-pressure process traces back to a first forwarding node, if this node cannot find a feasible next-hop node by itself, it will broadcast a **EPN** or **EPND** to ask help from its neighboring node. Upon receiving an **EPN** packet, a node proceeds as follows.

- 1) Checks if there is corresponding data packet in its data buffer. If not, it drops the packet. Otherwise, it goes to Step 2).
- 2) Changes the forwarding vector to the one from itself to the target, records the transmission status of the packet as **CENTER-FORWARDED** and forwards the packet.
- 3) Sets a timer to the void-avoidance delay, then listens the channel and records the position information of the forwarding node if any.
- 4) After void-avoidance delay, if the node finds out it is void node, it broadcasts an **EPN** and marks the transmission status of the packet as **TERMINATED**.

VBVA processes an **EPND** the same way as an **EPN** except that the receiving node extract data packet directly from the **EPND**.

Fig. 4 show us the status transition of a packet in our VBVA protocol.

From the above, we can clearly see that if a node receives a **VS**, **VSD**, **EPN**, or an **EPND** packet, then the node potentially can forward the data packet with a new forwarding vector from itself to the target. This might results in several forwarding vectors for one data packet, which will degrade the system's energy efficiency. To reduce the energy consumptions of VBVA, we propose a new self-center adaption.

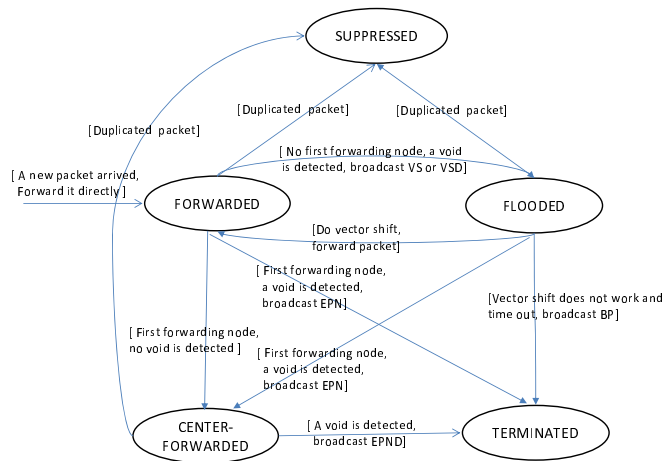


Fig. 4. Status transition diagram of a packet

G. Protocol Enhancement: Self-Center Adaptive Algorithm

In VBVA, when a node is qualified to initialize a new vector from itself to the target for a packet, the node delays the process for some time period, called self-center delay, which is determined by the positions of the receiving node and the requester¹. After self-center delay, if the node does not overhear any transmission of this packet which means that the node is at the best position to initiate a new forwarding vector, the node forwards the packet with a new forwarding vector. However, if this node overhears the transmission of the same packet from other nodes during its delay, it stops its process to initiate a new forwarding vector and marks the transmission status of the data packet as SUPPRESSED.

The self-center delay is calculated as follows,

$$T_s = \sqrt{\beta} \times T_{delay} \quad (1)$$

where T_{delay} is the maximum delay window. And β is defined as the self-center factor

$$\beta = \frac{(R - D)}{R} + \frac{R - R \times \cos\theta}{R}, \quad (2)$$

where R is the maximum transmission range and D is the distance between the node and the vector from the requester to the target, θ is the angle between the vector from the requester to the target and the vector from the requester to the node. Basically, self-center factor is used to evaluate the position of the receiving node, i.e., if the receiving node is closer to the target and farther from the requester, the self-center factor of the receiving node is smaller, which contributes to a lower self-center delay. When a node confronts a void, VBVA tries to avoid the void by shifting the forwarding vector away from the void node as far as possible, meanwhile, close to the target as much as possible.

As shown in the Figure 5, node A and node B receive VSs from forwarding node F, node T is the target node. Since the self-center factor of node A is smaller than that of node B, node A would forward the data packet with a new forwarding vector \overrightarrow{AT} , once node B overhears the transmission of the data packet, node B just drops its VS.

¹Here, we call the sender of VSs, VSDs, EPNs or EPNDs requester.

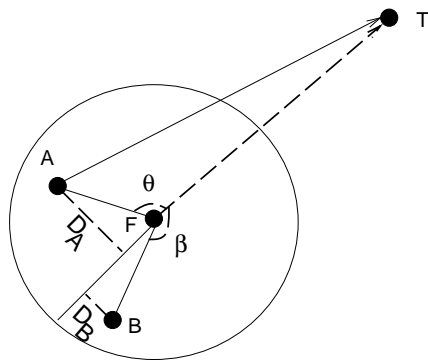


Fig. 5. An example for self-center factor

IV. PROTOCOL ANALYSIS

In VBVA, once a node detects a void for a packet, the node first avoids the void by vector-shift method. If the vector-shift method fails, then the node adopts back-pressure to route the packet backward. VBVA handles the void problem on demand. This means that VBVA is robust against mobile and dynamic voids since VBVA does not require any topology information before forwarding a packet.

VBVA has high end-to-end packet delivery ratio. If the underlying MAC is collision free and the topology of the network is stable during the time period of a packet delivery, VBVA can guarantee to find a path from the source to the target. And we have the following theorem.

Theorem 1: If the underlying MAC is collision free and the topology of the networks is connected and stable during a data transmission, VBVA can guarantee to find a path from the source to the target.

Before we prove the theorem 1, we need to define a *segment* of a forwarding path. A forwarding path is a acyclic sequence of nodes. The first node is the data source and the last one is the target. A segment is maximum part of the sequence of nodes such that all the nodes are in the forwarding pipe define by the first node of the sequence and the target node. For example, as shown in Fig. 6. $S_1S_2 \dots S_8$ a path from the source S_1 to target S_8 , where only the adjacent nodes are in the transmission range each other. $S_1S_2S_3$, S_4S_5 and $S_6S_7S_8$ are three segments respectively. It is easy to see that the minimum number of segment is 1 and the maximum number of segments in the path is the total number of nodes.

We prove that if there exist a path from the source to the target in the network, VBVA can find it.

Proof: We prove this by induction on the number of segments in the forwarding path.

We denote the acyclic path between the source and the target as $P_1P_2 \dots P_n$, where P_1 and P_n are the source and target respectively. First we prove that if the number of segments of the path from the source and the target is 1, then this path is included in the forwarding vector from the source to the target and VBVA can definitely find this path.

Let us assume that VBVA can find a path when the number of segments in the forwarding path is less than k . We now assume the path, $P_1P_2 \dots P_n$, has k segments. Let $P_1P_2 \dots P_i$ be the nodes in the first segment. There are two cases for the first segment $P_1P_2 \dots P_i$:

- 1) P_1, P_2, \dots, P_i are the nodes in the forwarding pipe defined by the vector from the source to the

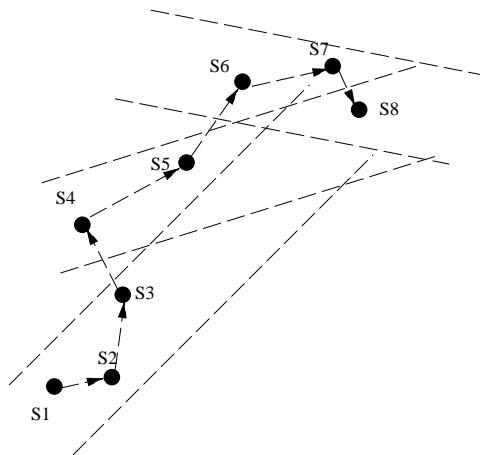


Fig. 6. An example of segments in a forwarding path

target. Therefore, P_i will receive the packet. There are two possibilities.

- a) Node P_i is a void node. If node P_i is the void node, by VBVA, P_i sends a **VS** or **EPN** if $i = 1$. Therefore, node P_{i+1} forwards the packet with the forwarding vector started from itself to the target. Since the number of segments $P_{i+1}P_{i+2} \dots P_n$ is $k - 1$, by our hypothesis, VBVA can route the packet from P_{i+1} to target P_n . Therefore, VBVA finds the path $P_1P_2 \dots P_n$.
 - b) Node P_i is not a void node, that means, there are still nodes in the forwarding pipe defined by $\overrightarrow{P_1P_n}$. Since there is no path to the target in the direction, a **BP** is eventually forwarded to node P_i , which is able to shift the forwarding vector to node P_{i+1} . Node P_{i+1} then forwards the packet with the forwarding vector started from itself to the target. Since the number of segments $P_{i+1}P_{i+2} \dots P_n$ is $k - 1$, by our hypothesis, VBVA can route the packet from P_{i+1} to target P_n .
- 2) P_1, P_2, \dots, P_i are the nodes in the space opposite the vector from the source to the target. By VBVA, the packet first is forwarded along the vectors from P_1, P_2, \dots, P_{i-1} to the target P_n respectively, however, since there is no path in these vectors, a **BP** is eventually forwarded to node P_i . Node P_i first shifts the forwarding vector to node P_{i+1} by broadcasting an **EPN** or **EPND**. Node P_{i+1} then forwards the packet with the forwarding vector started from itself to the target. Since the number of segments $P_{i+1}P_{i+2} \dots P_n$ is $k - 1$, by our hypothesis, VBVA can route the packet from P_{i+1} to target P_n .

Therefore, we can see that if there exist only one path in the topology, VBVA can find this path. ■

V. SIMULATION RESULTS

In this section, we evaluate the performance of VBVA based on simulations. We implement a simulator for underwater networks based on ns-2. The underlying MAC used here is a broadcast MAC protocol. In this MAC, if a node has data to send, it first senses the channel. If the channel is free, then the node broadcasts the packet. Otherwise, it backs off. The maximum number of backoffs is 4 for one data packet.

A. Parameter Settings

In all our simulations, we set the parameters similar to UWM1000 [14]. The bit rate is 10 kbps, and the transmission range is 100 meters. The energy consumption on sending mode, receiving mode and idle mode are 2 W, 0.75 W and 8 mW respectively. The size of the data packet and large control packet for VBF and VBVA in the simulation is set to 50 bytes. The size of the small control packet for VBVA is set to 5 bytes. The pipe radius in both VBVA and VBF is set 100 meters. The maximum delay window, T_{delay} of VBVA is set to 1 second.

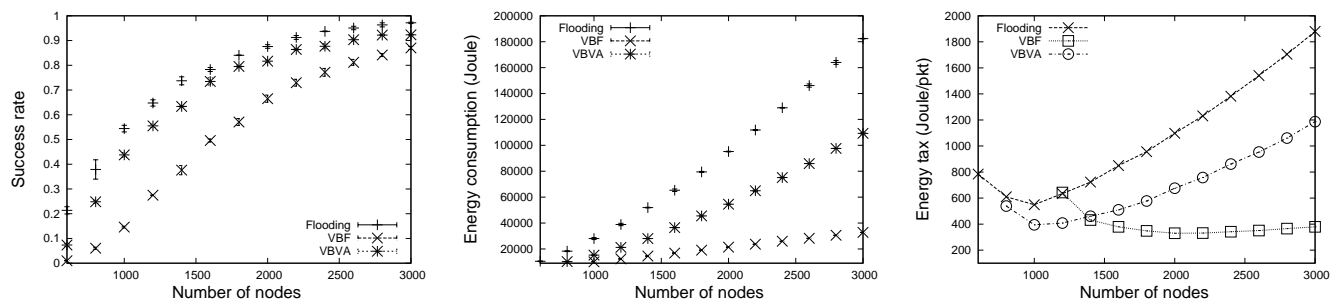
In all the simulation experiments described in this section, sensor nodes are randomly distributed in a space of $1000\text{ m} \times 1000\text{ m} \times 500\text{ m}$. There are one data source and one sink. The source sends one data packet per 10 seconds. For each setting, the results are averaged over 100 runs with a randomly generated topology. The total simulation time for each run is 1000 seconds.

Performance Metrics

We examine three metrics: success rate, energy cost and energy tax. The *success rate* is the ratio of the number of packets successfully received by the sink to the number of packets generated by the source. The *energy cost* is the total energy consumption of the whole network. The *energy tax* is the average energy cost for each successfully received packet.

B. Performance in random networks

In this simulation setting, the source is fixed at $(500, 1000, 250)$, while the sink is at located at $(500, 0, 250)$. Besides the source and sink, all other nodes are mobile as follows: they can move in horizontal two-dimensional space, i.e., in the X-Y plane (which is the most common mobility pattern in underwater applications). Each node randomly selects a destination and speed in the range of $0 - 3\text{ m/s}$, and moves toward that destination. Once the node arrives at the destination, it randomly selects a new destination and speed, and moves in a new direction. We compare the success rate of flooding, vector-based void-avoidance (VBVA) and vector-based forwarding (VBF). The data packet size in flooding is set to 40 bytes.



(a) Success rate: flooding, VBF and VBVA (b) Energy cost: Flooding, VBF and VBVA. (c) Energy tax: Flooding, VBF and VBVA.

Fig. 7. Comparison between Flooding, VBF and VBVA

From Figure 7(a), we can see that the success rate of VBVA is very close to that of flooding and much higher than that of VBF when the number of nodes is low. When the number of nodes in the networks

TABLE I
CONCAVE VOID VS CONVEX VOID

Void	Success rate	Energy tax (Joule/pkt)
Concave void	0.992600	606.450097
Convex void	0.99700	603.503211

increases, the probability of the presence of void is decreased, the difference among these three protocols is becoming less and less. When the network is very sparse, flooding algorithm and VBVA outperform VBF. Moreover, VBVA shows almost the same capability to overcome the voids in the networks as flooding. And the success rate of flooding roughly is the upper bound for all the routing protocols proposed for mobile networks.

As shown in Figure 7(b), VBF is the most energy efficient among these three protocols since VBF never attempts to consume more energy to overcome the voids in the networks. VBVA is energy efficient than flooding under all the network deployments. Notice that the difference between the energy cost of VBVA and flooding algorithm increases as the network becomes denser. This is attributed to the fact that VBVA overcomes voids with an energy efficient way. On the other hand, VBVA consumes more energy than VBF.

The energy taxes of the Flooding, VBF and VBVA are shown in Figure 7(c). From this figure, we can see that when the network is sparse, both flooding and VBVA consume less energy to successfully deliver one packet than VBF does. However, when the number of nodes exceeds 1200, flooding algorithm consumes more energy per packet than VBF and VBVA. VBVA consumes more energy per packet than VBF when the number of nodes in the network exceeds 1400. VBVA has lower energy tax than flooding algorithm under all the network topologies. Notice that after some point (1400), VBVA has higher energy tax than that of VBF, VBVA is still preferable over VBF for some application scenario where success rate is important. Since VBVA is based on VBF, it is easy to integrate VBVA as an option for VBF such that the application can determine if it is worth turning on void-avoidance.

From this simulation setting, we can see that VBVA approximates flooding algorithm on the success rate, but at much less energy cost. VBVA can avoid voids in mobile networks effectively and efficiently.

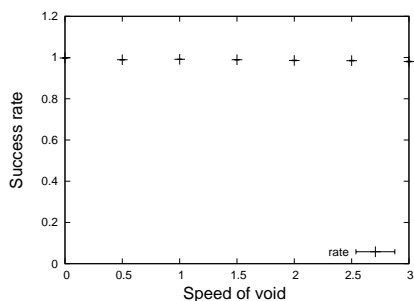
C. Handling Concave and Convex Voids

In this simulation setting, the target is fixed at location $(500, 0, 250)$ and the the source is fixed at location $(500, 1000, 250)$. We generate two different voids: concave and convex voids for the network. In order to generate concave and convex voids, we divide the whole space into smaller cubes $(50 \times 50 \times 50)$. A node is randomly deployed in each cube. We generate an ellipsoid centered at $(500, 500, 250)$ with radius $(300, 300, 150)$. The larger ellipsoid is the convex void in which there no nodes deployed. In order to generate a concave void, we generate another smaller ellipsoid centered at $(500, 300, 250)$ with radius $(200, 200, 150)$. These two ellipsoids overlap each other. The cubes in the overlapped part within the larger ellipsoid are deployed with sensor nodes, other cubes inside other parts of the large ellipsoid are empty. By doing this, we created an approximate concave void and the simulation results are shown in Table I.

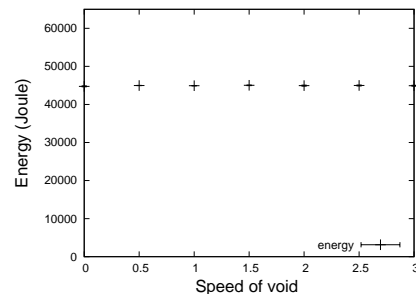
Table I demonstrates that VBVA can effectively bypass both the concave and the convex voids. From the table, we can see also that it costs more energy to bypass the concave void than convex void since there involves back-pressure mechanism in VBVA when it addresses the concave void. In section IV, we have proved that our VBVA can definitely find a route for a packet if the network is connected. Here, we can see that our VBVA can always achieve at least 99% success rate. only less than 1% packets get lost because no path exists from the source to the destination in the simulated random network.

D. Handling Mobile Voids

In this simulation setting, the target is fixed at location (500, 0, 250) and the the source is fixed at location (500, 1000, 250). In order to make sure there exist a path from the source to the target, we divided the whole space into small cubes (50 m \times 50 m \times 50 m). A sensor node is randomly deployed in each cube to guarantee that there exists a path from the source to the target. We can generate a sphere with the radius of 120 meters. The sphere moves back and forth along the straight line from the source to the target. All the nodes in the sphere are blank out, i.e., they can not receive and transmit any packets. The simulation results are shown in Figure 8(a) and Figure 8(b).



(a) Success rate: Flooding, VBF and VBVA.



(b) Energy cost: Flooding, VBF and VBVA.

Fig. 8. Performance in mobile networks

From Figure 8(a), it can be observed that VBVA can get almost 100% success rate under various mobile void speeds. Figure 8(b) shows that the mobility of the void has less effect on the energy cost of VBVA.

The simulations in this setting show that VBVA address mobile void efficiently and effectively. When the mobility speed of the void is less than 3 meters/s, the mobility of void has no effect on the success rate and energy efficient if there exists a path between the source to the target.

VI. CONCLUSIONS

The routing void problem in underwater sensor networks is characterized as three dimensional space, highly mobile nodes and mobile voids, which pose the most challenging problem for geographic routing protocols.

In this paper, we propose a void avoidance protocol, called vector-based void avoidance (VBVA) to address the void problem in mobile underwater sensor networks. VBVA is the first protocol to address the three-dimensional and mobile voids. VBVA adopts two mechanisms, vector-shift and back-pressure

to bypass voids in the forwarding path. VBVA detects a void only when needed and handles the void on demand. It does not require the topology information of the network and is very robust against mobile nodes and mobile voids.

We evaluate the performance of the VBVA under various network scenarios. The simulation results show that VBVA can handle both concave and convex voids as well as mobile voids effectively and efficiently.

REFERENCES

- [1] I. K. A. Goel, A. G. Kannan and R. Bartos. Improveing Efficiency of a Flooding-based Routing Protocol for Underwater Networks. In *Proceedings of the 3rd ACM international workshop on Underwater networks*, pages 91–94, Sep 2008.
- [2] I. F. Akyildiz, D. Pompili, and T. Melodia. Challenges for Efficient Communication in Underwater Acoustic Sensor Networks. *ACM SIGBED Review*, Vol. 1(1), July 2004.
- [3] 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.
- [4] S. Chen, G. Fan, and J. Cui. Avoid Void in Geographic Routing for Data Aggregation in Sensor Networks. *International Journal of Ad Hoc and Ubiquitous Computing, Special Issue on Wireless Sensor Networks*, Vol.2(1), 2006.
- [5] J. Cui, J. Kong, M. Gerla, and S. Zhou. Challenges: Building scalable mobile underwater wireless sensor networks for aquatic application. *IEEE network, Special Issue on Wireless Sensor Networking*, Vol.20(3):12–18, May/June 2006.
- [6] Q. Fang, J. Gao, and L. Guibas. Locating and Bypassing Routing Holes in Sensor Networks. In *Proc. of IEEE INFOCOM 2004*, Hong Kong, China, March 2004.
- [7] J. E. Garcia. Ad hoc Positioning for Sensors in Underwater Acoustic Networks. In *IEEE Proceedings of Oceans*, 2004.
- [8] Z. Guo, G. Colombo, B. Wang, J.-H. Cui, D. Maggiorini, and G. P. Rossi. Adaptive Routing in Underwater Delay/Disruption Tolerant Sensor Networks. In *Proceedings of the WONS*, pages 31–39, Jan 2008.
- [9] J. Heideman, W. Ye, J. Wills, A. Syed, and Y. Li. Research Challenges and Applications for Underwater Sensor Networking. In *IEEE Wireless Communication and networking Conference*, Las Vegas, Nevada, USA, April 2006.
- [10] J. M. Jornet, M. Stojanovic, and M. Zorzi. Focused Beam Routing Protocol for Underwater Acoustic Networks. In *Proceedings of the 3rd ACM international workshop on Underwater networks*, pages 75–81, Sep 2008.
- [11] B. Karp and H. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *MOBICOM'00*, Boston, MA, USA, August 2000.
- [12] J. Kong, J.-H. Cui, D. Wu, and M. Gerla. Building Underwater Ad-hoc Networks for Large Scale Real-time Aquatic Applications. In *IEEE Military Communication Conference (MILCOM'05)*, 2005.
- [13] N. H. Kussat, C. D. Chadwell, and R. Zimmerman. Absolute Positioning of an Autonomous Underwater Vehilce Using (GPS) and Acoustic Measurements. *IEEE Journal of Oceanic Engineering*, 30(1):153–164, Jan 2005.
- [14] LinkQuest. <http://www.link-quest.com/>.
- [15] M. Mauve, J. Widmer, and H. Hartenstein. A Survey on Position-based Routing in Mobile Ad-Hoc Networks. *IEEE Network Magazine*, 15(6):30–39, November 2001.
- [16] N. C. Nicolaou, A. G. See, P. Xie, J.-H. Cui, and D. Maggiorini. Improving the Robustness of Location-Based Routing for Underwater Sensor Networks. In *Oceans'07*, Aberdeen, Scotland, June 2007.
- [17] D. Pompili and T. Melodia. Three-Dimensional Routing in Underwater Acoustic Sensor Networks. In *Proceedings of the WASUN*, pages 214–221, Montreal, CA, Oct 2005.
- [18] J. Proakis, E.M. Sozer, J. A. Rice, and M. Stojanovic. Shallow Water Acoustic Networks. *IEEE Communications Magazines*, pages 114–119, November 2001.
- [19] J. G. Proakis, J. A. Rice, E. M. Sozer, and M. Stojanovic. *Shallow Water Acoustic Networks*. Ed. John Wiley and sons, 2001.
- [20] I. Stojmenovic and X. Lin. Loop-Free Hybrid Single-Path/Flooding Routing Algorithm with Guaranteed Delivery for Wireless Networks. In *IEEE Transactions on Parallel and Distributed Systems*, volume 12, Oct. 2001.
- [21] G. G. Xie and J. H. Gibson. A Network Layer Protocol for UANs TO Address Propagation Delay Induced Performance Limitations. In *Proceedings of the MTS/IEEE Oceans*, pages 1–8, Nov 2001.
- [22] P. Xie, J.-H. Cui, and L. Lao. VBF: Vector-Based Forwarding Protocol for Underwater Sensor Networks. In *Proceedings of IFIP Networking'06*, Coimbra, Portugal, May 2006.

- [23] Y. Xu, W.-C. Lee, J. Xu, and G. Mitchell. PSGR: Priority-based Stateless Geo-Routing in Wireless Sensor Networks. In *MASS'05*, Washington, D.C., USA, November 2005.
- [24] Y. Zhang and L. Cheng. A Distributed Protocol for Multi-hop Underwater Robot Positioning. In *IEEE Proceedings of International Conference on Robotics and Biomimetics*, August 2004.
- [25] Z. Zhou, J.-H. Cui, and A. Bagtzoglou. Scalable Localization with Mobility Prediction for Underwater Sensor Networks. In *UCONN CSE Technical Report:UbiNet-TR07-01*, July 2007.
- [26] Z. Zhou, J.-H. Cui, and S. Zhou. Localization for Large-Scale Underwater Sensor Networks. In *In Proceedings of IFIP Networking'07*, pages 108–119, Atlanta, Georgia, USA, May 14-18 2007.