

VIGOR+: Energy-Efficient Local Visibility Graph-based Geographic Routing in Large-Scale Sensor Networks

Myounggyu Won and Radu Stoleru

Department of Computer Science and Engineering, Texas A&M University
{mgwon, stoleru}@cse.tamu.edu

Abstract—Geographic routing is well suited for point-to-point communication in large-scale wireless sensor networks, because it is a stateless protocol - that is, the per-node state is independent of network size. However, the local minimum caused mainly by network holes makes the path stretch increase by up to $O(c^2)$, where c is the optimal path length. Recently, Tan et. al. proposed a visibility graph-based geographic routing protocol (VIGOR) that bounds the path stretch by a constant. VIGOR scales well, because its per-node state does not depend on network size. However, the scalability and bounded path stretch are possible at the cost of high communication overhead, which makes the employment of the protocol in a practical environment challenging. In this paper, we propose VIGOR+ that reduces the overhead of VIGOR by dividing its visibility graph into subgraphs, each representing the internal structure of a hole in a network; VIGOR+ uses visibility graph-based routing only within each subgraph; the routing outside subgraphs is handled by a novel scheme that uses the convex hull representation of a hole, a much reduced data structure. Through extensive simulations, we show that VIGOR+ has an average path stretch close to 1, while reducing the communication overhead as high as two orders of magnitude in our network configurations.

I. INTRODUCTION

Geographic routing has received significant attention from the wireless sensor network (WSN) research community due to its attractive properties: first, it is simple - that is, a forwarding node sends a packet to the geographically closest neighbor to a destination, without relying on control packet propagation [1][2], or beacon node selection [3]; second, geographic routing is highly scalable, because its per-node state is independent of network size. These properties allow geographic routing protocols to play a key role in many applications for WSNs such as disaster response [4], medical care [5], and target tracking [6] among others.

Geographic routing, however, fails when a forwarding node has no neighbor closer to the destination. This point of routing failure is called the *local minimum*. The local minimum is often caused by complex topological structures such as *network holes*. A number of protocols [7][8][9][10] have proposed to address the local minimum problem; the most notable and widely used algorithm is the face routing protocol that uses the faces of the planarized network topology to route a packet out of the local minimum [11]. Using face routing to address the local minimum, however, leads to increased path stretch. Kuhn et. al. [12] proved that the path stretch of geographic routing can increase by up to $O(c^2)$, where c is the optimal path stretch. Thus, achieving a constant stretch for geographic

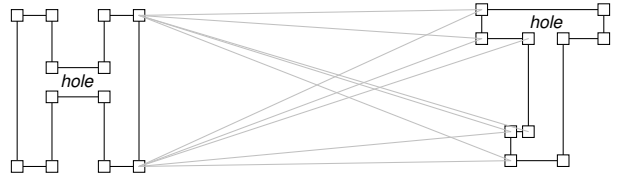


Fig. 1. Illustration of two holes abstracted as VON polygons (a VON node is represented as a small square) and communication overhead of VIGOR. The gray-colored line segments represent communication edges between VON nodes on the two VON polygons.

routing protocols has remained an important problem.

Recently, Tan et. al. [13] proposed a visibility graph-based geographic routing protocol (VIGOR) that achieves a constant path stretch. In this protocol, a hole is concisely represented as a polygon, called a VON polygon, as shown in Figure 1. The VON nodes (i.e., the vertices of the VON polygons) and the edges between VON nodes that are visible with each other form a visibility graph. This visibility graph serves as an overlay network on which a distance vector routing algorithm is run to find a shortest path. VIGOR, however, has some critical limitations. First, given a pair of source and destination nodes, *an additional control packet transmission*, from the source to the destination is required, to allow them to join the visibility graph. Source nodes use a default geographic routing protocol (e.g., GPSR or GOAFER+) to send this control packet to destination nodes. If we take the control packet transmissions into account, the path stretch of VIGOR can no longer be considered constant. Furthermore, VIGOR requires periodic *communication for building routing tables* to be used by the distance vector routing. More precisely, VON nodes must exchange messages with neighboring VON nodes until their routing tables converge; if two VON nodes are distant from each other, this communication overhead can become more significant, as illustrated in Figure 1.

In this paper, we propose an energy efficient visibility graph-based geographic routing protocol, called VIGOR+, that significantly reduces the communication overhead of VIGOR, while maintaining the guaranteed path stretch property. Our protocol is inspired by the observation that, if a source and destination pair is *outside the convex hulls* of VON polygons, representing holes as convex hulls is sufficient for finding a path with constant stretch. This observation allows us to use a visibility graph only when we route packets to/from nodes *inside convex hulls*. Thus, we can reduce the commu-

nication overhead for building routing tables by decomposing the global visibility graph into subgraphs. In our scheme, a subgraph, called a *local visibility graph*, is created for each VON polygon. Control messages for building routing tables are only locally exchanged among VON nodes of the same VON polygon, and not between distant VON nodes, that belong to different VON polygons. In addition, to eliminate the control packet communication overhead for setting up a routing path for VIGOR, our protocol strategically switches between two routing modes, namely *outside-convex routing* and *inside-convex routing*, depending on where forwarding and destination nodes reside. We develop a forwarding algorithm for outside-convex routing that produces a path with a constant stretch without relying on the use of a control packet. Nodes use the outside-convex routing as a default forwarding scheme; when forwarding nodes find that either a source or a destination nodes, is inside a convex hull, they switch the routing mode to inside-convex routing. We develop a *gateway-based* forwarding algorithm for the inside-convex routing mode, that enables forwarding nodes to either route a packet out of a convex hull, or to deliver a packet to nodes inside a convex hull, without relying on additional control packet transmissions. We prove that the inside-convex routing also provides a guaranteed path stretch.

In summary, the contributions of our paper are as follows:

- We identify that the state of art constant path stretch geographic routing protocol incurs significant communication overhead, which affects its practical deployment.
- We propose a local visibility graph-based geographic routing protocol (VIGOR+) that significantly reduces the communication overhead of the state of art solution.
- Through extensive simulations, we show that VIGOR+ has an average path stretch close to 1, while exhibiting energy and storage efficiency compared with the state-of-art solution.
- Through theoretical analysis, we show that the path stretch of VIGOR+ is bounded by a constant in a given network.

We review the state of art in Section II and introduce preliminaries in Section III. We present our local visibility graph-based geographic routing in Section IV and its performance evaluation in Section V. We conclude in Section VI.

II. RELATED WORK

It is known that hierarchical routing protocols and geographic routing protocols are well suited for large-scale wireless sensor networks [3]. The state-of-art hierarchical routing protocol, called S4, was recently proposed by Mao et. al. [3][14]. This protocol has a worst-case path stretch of 3 and requires per-node state of order $O(N)$, where N is the total number of nodes. Although S4 has relatively small per-node state, the per-node state still depends on network size. Compared with hierarchical routing protocols, geographic routing protocols are stateless, thereby being highly scalable. However, a drawback of geographic routing protocols is that, due to the local minimum caused by network holes, their path

stretch can increase by up to $O(c^2)$, where c is the optimal path length [12]. Achieving a constant path stretch for geographic routing protocols has remained an open research question.

A number of protocols have been proposed to efficiently avoid a local minimum. Fang et. al. [8] developed a TENT rule that nodes use to determine whether they are at a local minimum. Arad et. al. [10] identifies the nodes at a local minimum by using the angle between two adjacent neighbors. However, this heuristic approach causes frequent failures of the algorithm. Liu et. al. [9] addresses the problem by dividing a network into k regions, and allowing each node to maintain a vector of size k , where the i -th element of the vector indicates whether this node is at a local minimum with respect to the i -th region. These protocols, however, are not significantly different from the face routing protocol, suffering from the “late reaction problem” [15]; that is, a routing path is corrected only when a packet reaches a local minimum, thereby increasing path stretch.

Some protocols propose to use non-local information to address the late reaction problem and ultimately improve the path quality. Jiang et. al. [7] introduced the notion of “unsafe area” that contains a node at a local minimum, and some neighbors of such node. Nodes in an unsafe area are notified of the existence of the void, so that such nodes can make an early detour around the void before a packet reaches the local minimum. Li et. al. [15] introduced a routing protocol that abstracts the information of a hole (i.e., the size and shape of a hole) as an ellipse. The information is then broadcast to nodes within h hops from the hole, and used by the nodes to avoid the hole. Although these protocols succeeded in improving the path stretch, they do not offer a guaranteed path stretch.

Recently, Tan et. al. introduced VIGOR [13], the first geographic routing protocol that achieves a constant path stretch. This protocol concisely represents each hole in a network as a polygon, called a VON polygon. VIGOR then builds a *visibility graph* containing the VON nodes and the links between those VON nodes that are visible with each other. VIGOR achieves a constant stretch by using the visibility graph as a backbone network, and running a distance vector routing algorithm on the backbone network. However, the property of the constant path stretch is possible at the cost of high communication overhead. Each VON node must iteratively exchange messages with other VON nodes until a complete routing table is built. A more critical part is the use of a control packet for each pair of source and destination node. Our proposed protocol VIGOR+ addresses the limitations of VIGOR.

III. PRELIMINARIES AND PROBLEM FORMULATION

This section introduces terms, notations, and definitions used throughout this paper, and formally describes the objectives we address in this paper. We consider a wireless sensor network in a two dimensional area with N nodes, denoted by a set $V = \{v_1, v_2, \dots, v_N\}$. We assume that nodes have a unique ID. Nodes are localized by using either an onboard GPS, or some node localization protocol [16]. A



Fig. 2. Illustration of local visibility graphs for holes H_1 and H_2 . Polygons with dotted lines represent the convex hulls of the holes. In LVG-G, edges between two VON nodes in different holes are not considered.

source node knows the location of the destination by using a location service [17], or hash-functions in a data centric storage scheme [18]. We denote a source node by s and a destination node by t .

We assume that there are k network holes, $\{H_1, H_2, \dots, H_k\}$ in the network. Each hole H_i is represented as a polygon P_i . This polygon is called a VON polygon. Let a set $P_i = \{p_1, p_2, \dots, p_n\}$ denote the VON polygon, where each element p_i refers to a VON node that represents a vertex of the VON polygon. A set $C_i = \{c_1, c_2, \dots, c_n\}$ denotes the convex hull of VON polygon P_i , where c_i is an extreme point of the convex hull (i.e., $C_i \subseteq P_i$). In particular, given a $s - t$ pair, we define an interfering hole as follows:

Definition 1: An **interfering hole** is the convex hull of a VON polygon that intersects a line segment st .

Before we define the visibility graph, we first define the “visibility” between two VON nodes as follows:

Definition 2: Given two VON nodes p_i and p_j , they are **visible** with each other if and only if the open line segment connecting p_i and p_j does not intersect any VON polygon P_i in a network. \square

A visibility graph is defined as follows:

Definition 3: A **visibility graph** is a graph $G_{vis} = (V_{vis}, E_{vis})$, where the vertex set $V_{vis} = \bigcup_{i=1}^k P_i$, and the edge set $E_{vis} = \{(p_i, p_j) : p_i \text{ and } p_j \text{ are visible; } p_i, p_j \in V_{vis}\}$. \square

The Euclidean distance between two nodes v_i and v_j is denoted by $d(v_i, v_j)$; especially, $d(v_i, \overline{ab})$ refers to the minimum Euclidean distance from node v_i to line segment \overline{ab} . We denote the path between two nodes v_i and v_j by $v_i \sim v_j$. The length of a path $|v_i \sim v_j|$ is defined as the sum of Euclidean distances between two adjacent nodes along the path; for example, the length of path $v_1 - v_2 - v_3$ is $\sum_{i=1}^{i=2} d(v_i, v_{i+1})$.

Having defined all terms, we can formally describe the visibility graph-based geographic routing (VG-G), the main idea of VIGOR.

Definition 4: Given any two nodes, say source s and destination t , the **visibility-graph based geographic routing (VG-G)** is to identify a path $s \sim t$ such that $|s \sim t|$ is minimized on the augmented visibility graph defined as the following: the augmented visibility graph is a graph $G'_{vis} = (V_{vis} \cup \{s, t\}, E'_{vis})$, where E'_{vis} is a set of edges taking the two new vertices s and t into account. \square

An important issue for VG-G is that, in order to build the

augmented visibility graph for each pair of s and t , source node s must send a control packet to destination node t . The use of such control packet incurs huge communication overhead; thus, our first objective is to define a solution that does not rely on using the augmented visibility graph. Furthermore, it is important to make the size of the visibility graph as small as possible in order to afford using a distance vector routing (e.g., the DSDV) on the visibility graph; thus, we consider decomposing the original visibility graph into multiple sub-visibility graphs, called *local visibility graphs*, to provide a more practical solution for resource constrained wireless sensor networks. Figure 2 shows an example of local visibility graphs. We formally define the *local visibility graph* as follows:

Definition 5: A **local visibility graph** for a hole H_i is a graph denoted by $G_{vis,i} = (P_i, E_{vis,i})$, where $E_{vis,i} = \{(u, v) : u, v \in P_i; u \text{ and } v \text{ are visible.}\}$ \square

Now we are ready to formally describe our objective, namely the *local visibility graph based geographic routing (LVG-G)* as follows:

Definition 6: Given a source s and destination t , the **local visibility graph-based geographic routing (LVG-G)** identifies a path $s \sim t$ such that $|s \sim t|$ is minimized by using only the local visibility graphs for holes in a network, without relying on an augmented visibility graph. \square

VIGOR+ implements technical details to realize LVG-G.

IV. ENERGY EFFICIENT VISIBILITY GRAPH-BASED GEOGRAPHIC ROUTING (VIGOR+)

This section first presents an overview of VIGOR+, while following subsections explain the details of each component of VIGOR+.

A. Main ideas

The VIGOR+ protocol consists of four main components: boundary node detection, VON polygon construction, routing table set-up, and data forwarding. The first two components (i.e., the boundary node detection and the VON polygon construction) closely follow the algorithms used in VIGOR [13]; our main contributions lie in the last two components. Before we proceed with the description of our protocol, we need to precisely define a hole. In a planarized network, each face is surrounded by multiple edges. The size of a face is defined by the number of such edges. Tan et. al. [13] defines a hole as a face for which the number of edges is greater than a predefined threshold ω , a system parameter. Other faces having fewer edges than the threshold are called small faces, which are considered to have small impact on the performance of geographic routing.

The boundary node detection component identifies nodes that are located along the edges of big faces. We call such nodes *boundary nodes*. As part of the identification of boundary nodes, this component finds a leader for each big face. In the VON polygon construction component, a leader for each hole initiates a probing packet that travels along the loop of boundary nodes to identify the VON nodes of a hole. Once all

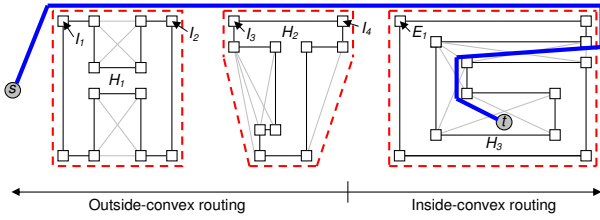


Fig. 3. Illustration of data forwarding.

the VON nodes of a hole are discovered and stored at a leader node, the leader broadcasts the locations of the VON nodes to nodes within h hops from the hole. Note that this localized distribution within h hops is not possible for VIGOR, because VIGOR requires all nodes in the network, to know about all the locations of VON nodes to build a global visibility graph.

Motivated by the fact that when two end points are not inside the convex hulls of VON polygons the convex hull information is sufficient to route a packet to the target end point along a path with guaranteed constant stretch, we decompose the visibility graph into multiple subgraphs such that each subgraph represents the internal structure of a convex hull (i.e., the convex hull of a VON polygon), and apply visibility graph-based routing only inside convex hulls. Accordingly, in the routing table set up phase, VON nodes build a routing table for local visibility graphs, by exchanging its routing table with neighboring VON nodes of the same VON polygon.

The data forwarding algorithm is the last component of VIGOR+. This algorithm focuses on the elimination of VIGOR's control packets for setting up a routing path. More precisely, this algorithm provides two routing modes, namely *outside-convex routing* and *inside-convex routing*. Forwarding nodes use the outside-convex routing as a default routing mode. Based only on the convex hulls of VON polygons, this routing mode allows a packet to reach a target end point on a path with a guaranteed path stretch without relying on the use of a control packet. Specifically, given source s and destination t , outside convex routing determines a set of intermediate destinations. For example, in Figure 3, I_1, I_2, I_3 , and I_4 are such intermediate destinations. If a source is inside a convex hull, or forwarding nodes find that destination t is inside a convex hull, our forwarding algorithm switches its mode to inside-convex routing, so that it either guides a packet out of a convex hull (i.e., when s in convex hull), or delivers a packet to a destination inside a convex hull without using a control packet (i.e., when t in convex hull). For example, intermediate destination I_4 , upon receiving a packet, changes the mode to inside-convex routing, and finds an entry point E_1 , preparing for the routing over the local visibility graph for hole H_3 . Upon receiving a packet, the VON node at E_1 routes the packet along the optimal path to destination t according to the distance vector routing on the local visibility graph.

B. Boundary detection

As explained in Section IV-A, boundary nodes for a hole must be identified to build a corresponding VON polygon.

VIGOR+ adopts the mechanism for finding boundary nodes from [13]. In this section, we briefly review the algorithm for identifying boundary nodes.

In planarized graphs, nodes are adjacent to a set of faces. A basic idea is to allow nodes to send a probing packet that travels along the edges of each adjacent face. This is done by geographically forwarding a packet to a fictitious point within the angle towards a target face. This packet then travels along the face according to the right-hand rule [11]; as it travels, it counts the number of edges that it has traveled. By our assumption, nodes have a unique ID. If the packet reaches a node with larger ID, the packet is dropped. Consequently, the probing packet returns to a node with the largest ID, which is called the leader node for the corresponding face. A leader node then checks the size of its face that is contained in the returned probing packet. If a leader determines that its face is a hole (i.e., the size of the hole is greater than predefined threshold ω), it initiates the next phase, the VON polygon construction; otherwise, it just drops the packet.

C. VON polygon construction

Once the Boundary Detection phase is finished, based on identified boundary nodes, leader nodes start constructing VON polygons for their holes as follows. A leader node becomes the first VON node. It then sends a probing packet that travels its boundary nodes in a clockwise direction. While the packet traverses boundary nodes, subsequent VON nodes are identified; more precisely, this probing packet contains three fields: the first field stores the locations of VON nodes identified so far; the second field contains the locations of visited boundary nodes since the most recently identified VON node; the third field has parameter δ that specifies the width of a bounding box used to identify VON nodes. Upon receiving the probing packet, a boundary node appends its location to the second field of the packet and checks whether there exists a bounding box of rectangular shape with width δ and unconstrained length, which can cover all the locations in the second field. If such a bounding box exists, boundary nodes forward the packet to the next boundary node; otherwise, the current boundary node is identified as a VON node – the location of this node is appended to the first field of the packet; the second field is emptied; and the packet is forwarded. We adopt the mechanism to eliminate possible intersecting edges among VON polygons from Tan et. al. [13].

D. Routing table set-up

After the VON polygon construction phase, VIGOR+ builds an overlay network by strategically connecting VON nodes. This overlay network serves as a backbone network on which nodes run a distance vector routing to find shortest paths. Recall that VIGOR implements the global visibility graph over this backbone network. However, this method incurs high communication overhead; thus, the main objective of this phase is to reduce the communication overhead by decomposing the global visibility graph into local visibility graphs.

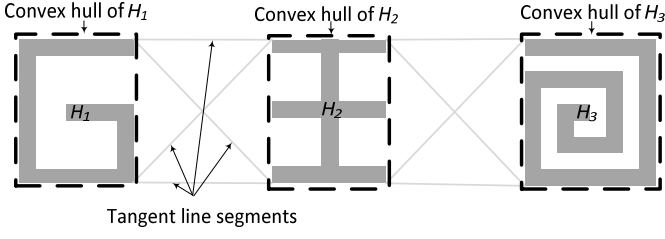


Fig. 4. Illustration of outside-convex routing.

When we finish the VON polygon construction phase, a leader node for each hole has a set of VON nodes. Leader nodes then broadcast the locations of their VON nodes to nodes within h hops from their VON polygon. Once VON nodes are distributed in the network, VON nodes start building their routing tables based on a distance vector routing protocol; specifically, VON nodes repeatedly exchange their routing table entries with neighboring VON nodes (i.e., visible VON nodes) in the same convex hull until their routing table stabilizes, forming a local visibility graph for each hole. Note that VON nodes refrain from exchanging routing tables with distant VON nodes that belong to different holes, to reduce the communication overhead.

E. Data forwarding

This section explains the details of the data forwarding component. To enable routing among local visibility graphs, this component supports two routing modes: *inside-convex routing* and *outside-convex routing*. We first describe the details of the outside-convex routing.

1) *Outside-convex routing*: Outside-convex routing is used to route a packet between two end points that are outside convex hulls. This routing makes a locally optimal routing decision based on the convex hulls of VON polygons within h hops; it refines the routing path as it guides a packet closer to the destination. Its operation can be summarized as three steps described as follows.

Step 1 (Identification of Interfering Holes): In this step, outside-convex routing identifies interfering holes. If there is any interfering hole, it proceeds to the next step; otherwise, it forwards a packet using a greedy forwarding.

Step 2 (Locally Optimal Routing): Once interfering holes are identified, outside-convex routing makes a locally optimal routing decision. More precisely, outside-convex routing considers possible paths connecting the two end points as follows. The convex hulls for interfering holes are connected via tangent line segments as shown in Figure 4. The combination of such tangent line segments gives all possible paths. The outside-convex routing chooses among them the shortest path; it then sets the first extreme point of the selected path as an intermediate destination. According to Rohnert's algorithm [19], such tangent line segments can be computed in $O(n + c^2 \log n)$, where c is the number of interfering holes, and n is the total number of vertices of the convex hulls. Moreover, it is important to note that the space complexity of

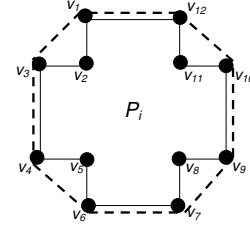


Fig. 5. An example showing the inner hole and gateways.

this step is just $O(n + c^2)$. We believe that, considering much smaller number of interfering holes compared to entire holes in a network, the space and processing complexities are close to the linear of n . Such low space and processing complexity due to reduced data structure (i.e., convex hulls) allows for the local computation of this step in a sensor node.

Step 3 (Routing to Intermediate Node): Outside-convex routing then checks whether there exists any interfering hole for a path connecting a source and the first intermediate destination. If there is no interfering hole, the packet is forwarded to the intermediate destination using simple geographic forwarding. Otherwise, Step 1 and Step 2 are rerun, where new intermediate destinations are found for a path between a source and the first intermediate destination. It is important to note that the number of such reruns in Step 3 is bounded.

Lemma 1: Step 3 is bounded.

Proof: Please see Appendix. \square

Using Lemma 1, we prove the constant path stretch property of the outside-convex routing.

Lemma 2: The path stretch of outside-convex routing is bounded by a constant.

Proof: Please see Appendix. \square

2) *Inside-convex routing*: Forwarding nodes switch to the inside-convex routing when they find that they are inside a convex hull, or that a destination is inside a convex hull. We first explain the case when a destination is inside a convex hull. In order to understand the inside-convex routing algorithm, we need to define several terms as follows.

Definition 7: A j -th **inner hole** of VON polygon P_i is a set of sequentially ordered VON nodes denoted by $P_{ij} = \{v_1, \dots, v_n\}$, where $P_{ij} \subset P_i$ such that v_1 is the first j -th extreme point of C_i , and v_n is the next extreme point of C_i . \square

Definition 8: Vertices v_1 and v_n of inner hole P_{ij} are called **gateway nodes**. \square

Figure 5 shows an example that illustrates four inner holes and four gateway pairs. The VON polygon $P_i = \{v_1, \dots, v_{12}\}$ is represented as a convex hull $C_i = \{v_1, v_3, v_4, v_6, v_7, v_9, v_{10}, v_{12}\}$; This VON polygon P_i has four inner holes; that is, $P_{i1} = \{v_1, v_2, v_3\}$, $P_{i2} = \{v_4, v_5, v_6\}$, $P_{i3} = \{v_7, v_8, v_9\}$, and $P_{i4} = \{v_{10}, v_{11}, v_{12}\}$. The gateways for P_{i1} , P_{i2} , P_{i3} , and P_{i4} are $\{v_1, v_3\}$, $\{v_4, v_6\}$, $\{v_7, v_9\}$, and $\{v_{10}, v_{12}\}$, respectively.

We consider a scenario where node u located outside convex hulls sends a packet to destination node t that is inside convex

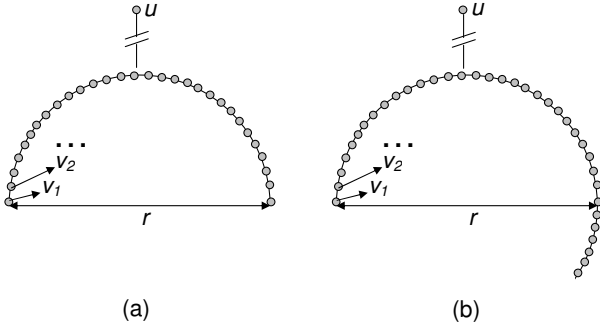


Fig. 6. Illustration for bounded stretch for inside-convex routing.

hull C_i . Convex hull C_i may have a set of gateways depending on the complexity of the hole P_i and the location of t . The objective of node u is to choose a VON node $v \in P_i$ from its visibility set (i.e., the set of visible VON nodes in P_i) such that $|u \sim v| + |v \sim t|$ is minimized. If u finds such v , u forwards a packet to v . After reaching v , the packet is transmitted along the shortest path to t following the local visibility graph for P_i . Our inside-convex routing algorithm serves this functionality.

The algorithm first finds the two gateways for inner hole P_{ij} that contains destination t . This can easily be computed because vertices of VON polygons are sequentially ordered; that is, we can find the edges of VON polygon P_i by sequentially accessing each vertex of P_i . More precisely, starting from one gateway (i.e., j -th extreme point of P_i), our algorithm sequentially stores the vertices of the VON polygon until another gateway (i.e., $j+1$ -th extreme point of P_i) is found. This set of vertices form the inner hole P_{ij} . We then apply a simple geometric algorithm [20] to find whether a given point (i.e., destination t) is inside this inner hole P_{ij} or not. If destination t is inside this inner hole, the two gateways are discovered (i.e., v_1 and v_n in P_{ij}).

Let a set U be the visibility set of node u that belongs to P_i , and let the two gateways denoted by g_1 and g_2 respectively. We must consider two cases separately: (1) both g_1 and g_2 are not in U ; and (2) one of g_1 and g_2 (or both g_1 and g_2) is in a set U . For the case (1), a packet must pass either g_1 or g_2 . Thus, we find $v \in U$ that minimizes $|u \sim v| + |v \sim t|$ as follows. We compute the two distances from each $v \in U$ to g_1 and g_2 respectively, along the edges of VON polygon P_i ; Let d_v denote the shorter distance between the two. We then choose $v \in U$ that satisfies $\min(d_v)$. The case (2) can be trickier. We cannot simply choose node $v \in U$ that is closer to either of gateways, because in this case, some nodes in U might be inside the inner hole that contains t . A problem is that nodes outside a convex hull do not know about the internal structure of the hole (i.e., the local visibility graph of the hole); thus, without sending a probing packet, it is impossible to optimally choose such $v \in U$. To tackle this case, we introduce a simple heuristic – node u chooses $v \in U$ that is geographically closest to t . We analyze that this heuristic method gives the stretch of a path $u \sim t$ bounded by $\frac{\pi \cdot r}{2}$.

Lemma 3: $|u \sim t| \leq \frac{\pi \cdot r}{2}$. Opt $|u \sim t|$, r is the diameter of

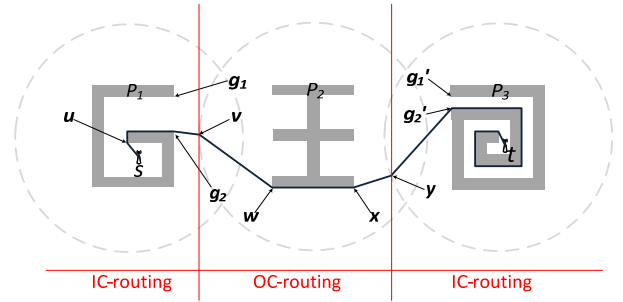


Fig. 7. An example for the forwarding algorithm.

a hole that contains destination t .

Proof: Consider n nodes, denoted by a set $U = \{v_1, v_2, \dots, v_n\}$, $U \subseteq P_i$ that are visible from node u . Our heuristic method allows node u to select $v \in U$ such that $d(u, v) + d(v, t)$ is minimized. Suppose that node v is not an optimal selection; that is, there exists v' for optimal choice. We are interested in the maximum distance between v and v' , which is the price that our heuristic method should pay for the wrong decision. Thus, the problem is to arrange nodes in U such that all n nodes are visible from u , and the distance $\sum_{i=1}^{n-1} d(v_i, v_{i+1})$ is maximized. Without loss of generality, assume that the diameter of a convex hull is r , and node u is sufficiently far from the convex hull. As illustrated in Figure 6(a), we can easily find that, when we arrange n nodes along the semicircle with radius r , $\sum_{i=1}^{n-1} d(v_i, v_{i+1})$ is maximized and all n nodes in U are visible from u . As shown in Figure 6(b), if we attempt to increase the distance by extending the semicircle, we end up with nodes that are not visible from u . \square

Now we explain the case where a source node is inside a convex hull. This case includes the scenario where both source and destination nodes are inside the same convex hull. Note that if source node s does not have interfering holes, t is the final destination; otherwise t is the first intermediate destination. Source node s can find the inner hole that it belongs to, as well as the two gateways of the inner hole. The objective of source node s is to choose a node v in its visibility set U that minimizes the distance $|s \sim v| + |v \sim t|$. Note that, after reaching such node v , a packet is transmitted out of the convex hull along the shortest path following the local visibility graph. However, source node s cannot make an optimal selection of such node v , because it does not have the knowledge on the internal structure of the hole; thus, we have to introduce a heuristic mechanism. This heuristic algorithm operates as follows. If node t is in U , source node s sends a packet to node t using simple geographic forwarding; otherwise, s chooses v in U such that d_v is minimized, where distance d_v is $|v \sim t|$ along the edges of P_i . It is easy to note that, using similar proof as in Lemma 3, this heuristic algorithm can also be shown to output a path $s \sim t$ with its stretch bound by $\frac{\pi \cdot r}{2}$.

Considering both the cases, we obtain the following.

Lemma 4: Inside-convex routing has a bounded stretch of

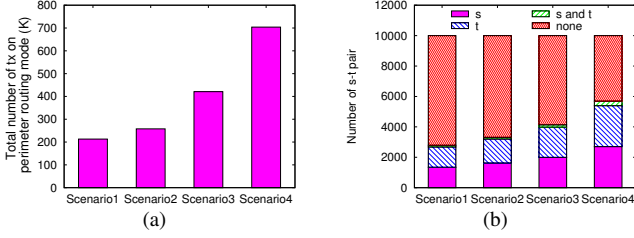


Fig. 8. (a) Total number of packet transmissions in perimeter-routing mode; (b) the number of $s-t$ pairs for which s , t , or both are inside convex hulls.

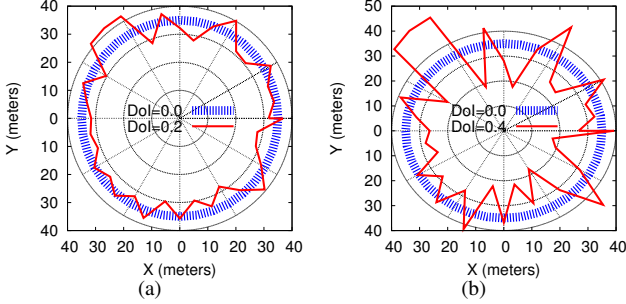


Fig. 9. (a) Radio range with DOI=0.2 (b) Radio range with DOI=0.4

$\frac{\pi \cdot r'}{2}$, where r' is the maximum diameter of holes.

For the combination of outside-convex and inside-convex modes, we can obtain the following result.

Theorem 1: VIGOR+ has a constant stretch in a given network.

Proof: Please see Appendix. \square

3) *Example:* Figure 7 shows an example describing the operation of our forwarding algorithm. Source node s first finds that it is inside a convex hull C_1 by checking received VON nodes. At the same time, source s does not identify interfering holes, because all other holes are more than h hops away from source s ; thus, source s just finds a visible VON node u that minimizes $|s \sim u| + |u \sim t|$. Once a packet reaches g_2 , default geographic routing is used to forward the packet to destination t . When the packet reaches v , it detects interfering hole C_2 , and starts the outside-convex routing, determining the intermediate points w and x . Upon reaching node x , the packet is then forwarded using a default geographic forwarding again. When a packet reaches y , it immediately detects that t is inside a convex hull, so that it changes its mode to inside-convex routing. The forwarding node y then identifies the node z located at g_2 such that $|y \sim z| + |z \sim t|$ is minimized. Upon reaching node z , the node uses a local visibility graph to guide the packet to destination t along the optimal path.

V. SIMULATION RESULTS

We implemented VIGOR+, VIGOR [13], GPSR [11], and a centralized shortest path routing protocol in C++, mainly focusing on the topological behavior of the protocols. We randomly deployed 6,000 sensor nodes in a network of $2,000 \times 2,000 \text{m}^2$ region. For this set of simulations, four different network scenarios were considered as shown in

Figure 10(a), 10(b), 10(c), and 10(d). Each scenario has a set of holes with different *complexity*. The *complexity* of a hole is measured as the total number of packet transmissions in the perimeter-routing mode when a packet is transmitted for 10,000 randomly chosen source and destination pairs. This complexity is depicted in Figure 8(a). As shown, the network scenarios are arranged in an increasing order of the complexity of deployed holes. This complexity directly relates to how often VIGOR+ changes its routing mode. Figure 8(b) shows that network scenarios with more complex holes have more source and destination nodes inside convex hull.

To account for the realistic communication channels, we employ the radio model from [21][22]. He et. al. [21] defines the *degree of irregularity* (DOI) as the maximum radio range variation in the direction of radio propagation. Figure 9(a) and Figure 9(b) show the radio range for DOI=0.2 and DOI=0.4, respectively.

We compared VIGOR with VIGOR+ and GPSR. Our main interests in this section are to show that VIGOR+ has significantly lower communication overhead compared with VIGOR, while VIGOR+ maintains as good path stretch as VIGOR. In particular, the comparison with GPSR is used to provide the base line performance. We measured the following metrics: (1) average path stretch, (2) maximum path stretch, and (3) communication overhead. The communication overhead consists of two parts – one part measures the total number of packet transmissions for setting up routing tables; and the other part measures the total number of packet transmissions for control packets to set up routing paths. We varied the following parameters: (1) h , (2) δ , and (3) DOI. The parameter h represents the number of hops from a hole, within which nodes receive the locations of VON nodes for the hole (i.e., the set of VON nodes). The parameter δ is the width of a bounding box used to find VON nodes.

Each simulation uses 10,000 randomly selected source and destination pairs. For each session, a source transmits a single packet to a destination. The default network configuration was: DOI=0.4, $\omega = 15$, $\delta = 30$, $h > 100$. The average degree of the network was approximately 11.

A. Path stretch

We measured path stretches for paths for 10,000 randomly selected source-destination pairs in each network scenario. We first define the path stretch as follows:

$$\text{path_stretch} = \frac{\text{measured_hop_count}}{\text{minimum_hop_count}}$$

Given a source and destination pair ($s-t$ pair in short), *measured_hop_count* refers to the number of hop counts of the path connecting source s and destination t ; and *minimum_hop_count* is the number of hop counts of the shortest path between s and t . This shortest path is computed by using a flood-based shortest path routing algorithm.

For each network scenario, we represent the distribution of path stretches of 10,000 samples as a cumulative distribution function (CDF). Figures 11(a), 11(b), 11(c), and 11(d) depict

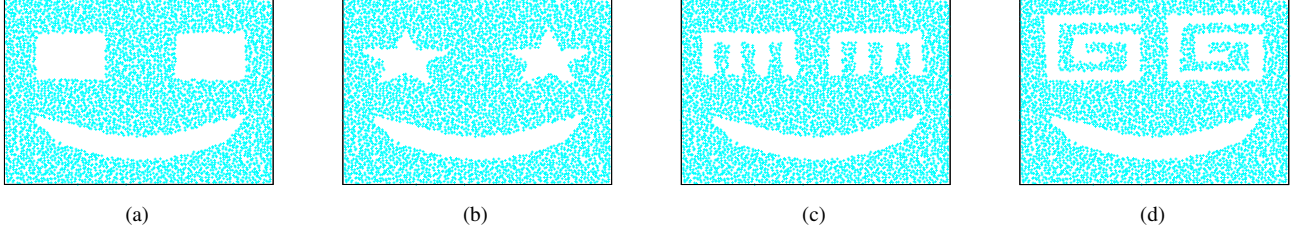


Fig. 10. Different hole deployment schemes: (a) hole Scenario 1; (b) hole Scenario 2; (c) hole Scenario 3; and (d) hole Scenario 4.

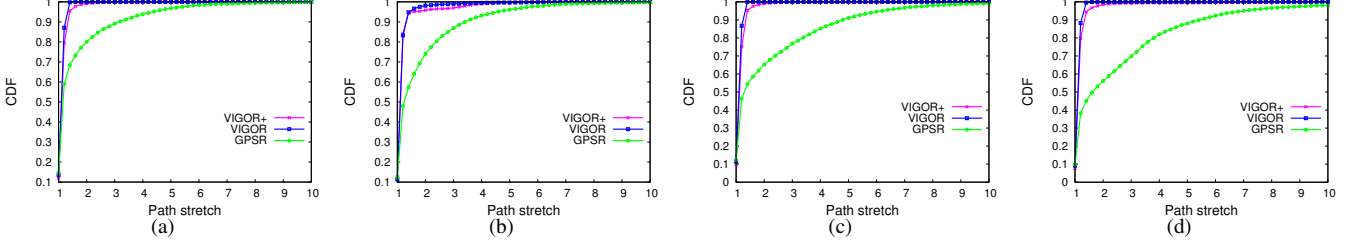


Fig. 11. The CDF graphs of path stretches for each deployment scenarios: (a) Scenario 1; (b) Scenario 2; (c) Scenario 3; and (d) Scenario 4.

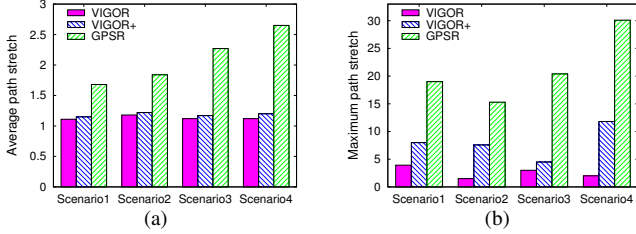


Fig. 12. (a) Average hop stretch; and (b) Maximum hop stretch.

the CDF for network scenario 1, 2, 3, and 4, respectively. As shown, regardless of the complexity of deployed holes, more than 90% of the path stretches for both VIGOR and VIGOR+ are very close to 1. The main reason is that both protocols use non-local information (i.e., the abstracted information on holes) to prevent a packet from falling into a local minimum, which usually leads to a high path stretch. Another observation is that the path stretch of VIGOR+ is slightly higher compared with VIGOR for the four network scenarios. This is because VIGOR+ makes a heuristic decision in determining the next hop when either source or destination is inside a convex hull, while VIGOR always make an optimal decision by using the control packet. However, as we will show in Section V-B, this control packet overhead poses a significant problem in large scale deployments. In contrast to the results of VIGOR and VIGOR+, GPSR is significantly affected by the complexity of holes; that is, as the complexity of holes increases from scenario 1 to scenario 4, the path stretch of GOAL degrades.

Average and maximum path stretches are statistical measurements often used to compare the performance of routing protocols. We summarize the samples of path stretches by means of the average and maximum path stretch to compare VIGOR+ with VIGOR and GPSR. Figure 12(a) shows the average stretches of the three routing protocols. We note

that, while the average path stretch of GPSR increases as the complexity of holes increase, both VIGOR and VIGOR+ show constantly low average path stretches close to 1. In particular, differences in the average path stretches of VIGOR and VIGOR+ are almost negligible. Figure 12(b) shows the maximum path stretches of the three routing protocols. As shown, we note that the maximum path stretch of VIGOR is unexpectedly much greater than 1, despite its optimal route selection. We found that this high maximum path stretch happens to paths with very small hop counts (e.g., 1 ~ 3); more precisely, even a small increase in a hop count results in high path stretch for such paths. We identified that irregular communication range, and small holes that are not considered by VIGOR were the main causes for such increases in a hop count. Another observation is that the maximum path stretch of VIGOR+ is relatively higher than VIGOR. The main reason is the use of heuristic forwarding algorithm used to forward a packet to/from a node inside a convex hull. It is paramount to note that unlike VIGOR+, for GPSR, the maximum path stretch is directly influenced by the complexity of holes (e.g., an arbitrarily large path stretch obtained by very long spiral in Figure 10(d)).

B. Communication overhead

This section evaluated how much communication overhead reduction VIGOR+ has, when compared with VIGOR. Figure 13(a) shows the communication overhead for the “routing table set-up” of both VIGOR+ and VIGOR. We first observe that the communication overhead for both protocols increases as the complexity of deployed holes increases. The reason is straightforward; that is, the network requires more VON nodes to represent more complex holes. We also note that, by decomposing the global visibility graph into sub-visibility graphs, we can significantly reduce the communication overhead; specifically, we can see that VIGOR+ reduces this type

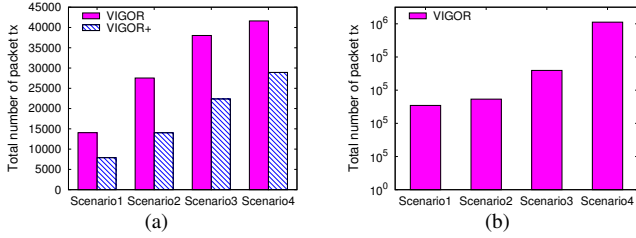


Fig. 13. (a) overhead for routing table set up; (b) overhead for routing path set up.

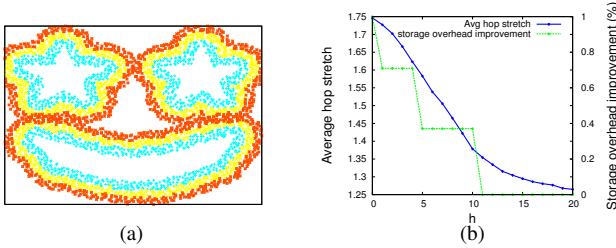


Fig. 14. (a) Nodes within h hops from holes; (b) Impact of h .

of communication overhead by up to 50%.

Figure 13(b) depicts the communication overhead for the “routing path set-up” of VIGOR. We measured this type of communication overhead for only unique $s - t$ pairs, because the same $s - t$ pair can use the previously found routing path. We note that this type of communication overhead for VIGOR increases as the complexity of deployed holes increase. The reason is that VIGOR uses its underlying default geographic routing protocol (i.e., GPSR or GOAFER+) to deliver the control packet for routing path set-up. As we noted in Section V-A, the path stretches of traditional geographic routing protocols are significantly influenced by the complexity of holes – therefore the increase in the communication overhead for higher complexity of holes. Also note that VIGOR+ does not suffer from this communication overhead, resulting in significant improvement in energy efficiency. In sum, considering both types of communication overhead, VIGOR+ had total of 28,920 packet transmissions, while VIGOR had 41,614 (overhead for routing table set up) + 1,010,000 (overhead for path set up) in Scenario 4, showing *improvements as high as two orders of magnitude*, when compared with VIGOR.

C. Impact of h

The parameter h in VIGOR+ specifies the number of hops from the boundary of a hole, within which nodes receive the set of VON nodes of the hole. For example, Figure 14(a) represents nodes within 2, 4, 6 hops from each hole using different colors, respectively. The parameter h is useful for large-scale networks with many holes, where storing all VON nodes in the entire network is infeasible; that is, parameter h allows nodes to store only partial information of deployed holes in the network. Furthermore, the set of VON nodes do not need to be flooded throughout the entire network. However, as Figure 14(b) shows, there is a tradeoff; that is, if

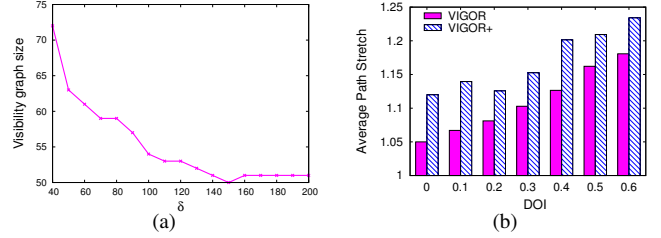


Fig. 15. (a) Impact of δ ; (b) Impact of DOI

we define the storage improvement as the percentage decrease of the storage overhead compared with VIGOR, tradeoffs exist between the improvement in the storage overhead and the average path stretch.

We measured the storage improvement and the average stretch of VIGOR+ for scenario 2. Figure 14(b) shows that, if we use smaller h values, the average path stretch of VIGOR+ increases. This is because nodes that are more than h hops away from the boundary of a hole do not know about the existence of a hole; thus, these nodes would forward a packet along the suboptimal routing path, resulting in the increase in the path stretch. However, the figure also shows that smaller h values allow nodes to improve the storage overhead, and the improvement decreases as parameter h increases. A reason is that, higher h values make nodes store information about more holes in a network.

D. Impact of δ

The parameter δ determines the width of the bounding box used to find VON nodes of a given hole. Using a bounding box with small δ values allows us to represent holes using more VON nodes; that is, holes are represented more precisely with smaller δ values. We measured the number of VON nodes (i.e., the size of the visibility graph) for network scenario 2 by varying the δ value. Figure 15(a) depicts the results. As shown, the size of the visibility graph significantly decreases with the first few decreasing δ values, and the decrease slows down with higher δ values. These results indicate that, if the δ value is sufficiently large, we can represent the visibility graph using a small number of VON nodes, leading to smaller communication overhead for routing table set-up for both VIGOR and VIGOR+. However, this is only possible with the cost of deteriorated path stretches of the protocol. The reason is that the small number of VON nodes results in the imprecise representation of holes, and the imprecise representation leads to more packet forwarding in perimeter routing modes. Furthermore, high δ values may cause crossing edges among VON polygons, which also contributes to the increased path stretches.

E. Impact of DOI

As we mentioned in Section V, higher DOI means more irregular communication ranges. This section is designed to see how irregular communication ranges affect the performance of VIGOR and VIGOR+. We measured the average path

stretches of both VIGOR and VIGOR+ by varying DOI values. Figure 15(b) shows the results. As shown, if we increased DOI from 0 to 0.6, the path stretches of both VIGOR and VIGOR+ increased by up to 12%. The reasons are simple: both VIGOR and VIGOR+ use a geographic routing protocol as its underlying routing protocol; when DOI values are high, it is more likely that geographic routing protocols select a wrong neighbor, which leads to a suboptimal routing path.

VI. CONCLUSIONS

We proposed VIGOR+ that achieves an average path stretch close to 1, while significantly reducing the communication overhead of VIGOR. VIGOR+ strategically divides the global visibility graph into subgraphs for each hole. Visibility-graph based routing is performed only within a subgraph, while a novel convex hull-based routing is used for forwarding packets outside subgraphs. Through extensive simulations, we demonstrated that VIGOR+ has as good path stretch as VIGOR, while reducing the communication overhead of VIGOR by as high as two orders of magnitude.

As our future work we will consider: a) the outer boundary of a network; more precisely, we plan to abstract the outer boundary as a set of VON polygons, and integrate the polygons into the current implementation of VIGOR+; and b) an implementation of VIGOR+ on real hardware; VIGOR+ was mainly designed for an emergency-response WSN application [4], where various types of holes exist in the deployment area. We plan to implement VIGOR+ on motes and test it in an indoor testbed and outdoors.

REFERENCES

- [1] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proc. of ACM MOBICOM*, 2000.
- [2] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proc. of IEEE Workshop on Mobile Computer Systems and Applications*, ser. WMCSA '99, 1999.
- [3] Y. Mao, F. Wang, L. Qiu, S. Lam, and J. Smith, "S4: Small state and small stretch compact routing protocol for large static wireless networks," *Networking, IEEE/ACM Transactions on*, pp. 761–774, June 2010.
- [4] S. M. George, W. Zhou, H. Chenji, M. Won, Y. Lee, A. Pazarloglou, R. Stoleru, and P. Barooah, "DistressNet: a wireless AdHoc and sensor network architecture for situation management in disaster response," *IEEE Communications Magazine*, vol. 48, no. 3, Mar. 2010.
- [5] D. Malan, T. Fulford-jones, M. Welsh, and S. Moulton, "Codeblue: An ad hoc sensor network infrastructure for emergency medical care," in *In International Workshop on Wearable and Implantable Body Sensor Networks*, 2004.
- [6] R. Brooks, P. Ramanathan, and A. Sayeed, "Distributed target classification and tracking in sensor networks," *Proc. of the IEEE*, vol. 91, no. 8, pp. 1163–1171, Aug. 2003.
- [7] Z. Jiang, J. Ma, and W. Lou, "An information model for geographic greedy forwarding in wireless ad-hoc sensor networks," in *Proc. of IEEE INFOCOM*, 2008.
- [8] Q. Fang, J. Gao, and L. J. Guibas, "Locating and bypassing holes in sensor networks," in *Proc. of IEEE INFOCOM*, 2004.
- [9] C. Liu and J. Wu, "Destination-region-based local minimum aware geometric routing," in *Proc. of IEEE MASS*, 2007.
- [10] N. Arad and Y. Shavitt, "Minimizing recovery state in geographic ad hoc routing," *IEEE Transactions on Mobile Computing*, vol. 8, 2009.
- [11] B. Karp and H. T. Kung, "Gpsr: greedy perimeter stateless routing for wireless networks," in *Proc. of ACM MOBICOM*, 2000.

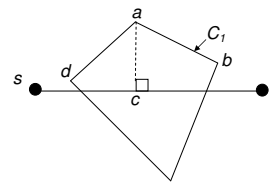


Fig. 16. Illustration of the height of an interfering hole.

- [12] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, "Geometric ad-hoc routing: of theory and practice," in *Proc. of ACM PODC*, 2003.
- [13] G. Tan, M. Bertier, and A.-M. Kermerrec, "Visibility-graph-based shortest-path geographic routing in sensor networks," in *Proc. of IEEE INFOCOM*, 2009.
- [14] Y. Mao, F. Wang, L. Qiu, S. S. Lam, and J. M. Smith, "S4: Small state and small stretch routing protocol for large wireless sensor networks," in *Proc. of USENIX NSDI*, 2007.
- [15] P. Li, G. Wang, J. Wu, and H.-C. Yang, "Hole reshaping routing in large-scale mobile ad-hoc networks," in *Proc. of IEEE GLOBECOM*, 2009.
- [16] R. Stoleru, J. Stankovic, and S. Son, "On composability of localization protocols for wireless sensor networks," *Network, IEEE*, vol. 22, no. 4, pp. 21–25, July-Aug 2008.
- [17] J. Li, J. Jannotti, D. S. J. De Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in *Proc. of ACM MOBICOM*, 2000.
- [18] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "Ght: a geographic hash table for data-centric storage," in *Proc. of ACM WSNA*, 2002.
- [19] Hans and Rohnert, "Shortest paths in the plane with convex polygonal obstacles," *Information Processing Letters*, vol. 23, no. 2, pp. 71–76, 1986.
- [20] I. E. Sutherland, R. F. Sproull, and R. A. Schumacker, "A characterization of ten hidden-surface algorithms," *ACM Comput. Surv.*, vol. 6, pp. 1–55, March 1974.
- [21] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, "Range-free localization schemes for large scale sensor networks," in *Proc. of ACM MOBICOM*, 2003.
- [22] L. Hu and D. Evans, "Localization for mobile sensor networks," in *Proc. of ACM MOBICOM*, 2004.

VII. APPENDIX

A. Proof of Lemma 1

For this proof, we need to define the height of an interfering hole as follows:

Definition 9: (Height of interfering hole). Given an interfering hole C_i for line segment \overline{st} , the height of interfering hole C_i is $\max_j d(j, \overline{st})$, where j is the intermediate destination for on C_i . The height of interfering hole C_i is denoted by $h(C_i)$. \square

For example, consider Figure 16. The interfering hole C_1 for line segment \overline{st} has three intermediate destinations (i.e., a , b , and d), among which intermediate destination a has the longest distance to line segment \overline{st} . This distance is the height of interfering hole C_1 .

Now consider some notations shown in Figure 17. The interfering hole for line segment \overline{st} is denoted by C_1 . We denote the first intermediate destination of C_1 by m_1 . Note that Step 3 of the outside-convex routing is repeated if there is an interfering hole for line segment $\overline{sm_1}$. This interfering hole is denoted by C_2 , and corresponding first intermediate destination on C_2 is denoted by m_2 , as shown in Figure 17. Depending on the deployment of holes, those extra interfering holes, other

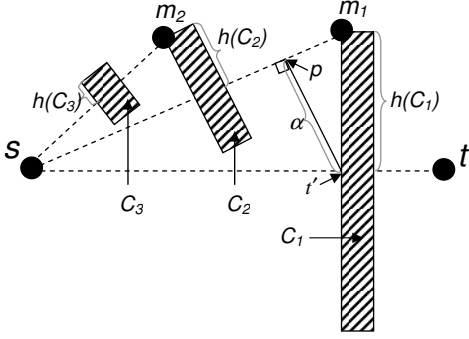


Fig. 17. Some notations for Lemma 1.

than C_1 for \overline{st} may be repeatedly found, i.e., there exists a sequentially ordered set $\mathcal{C} = \{C_1, C_2, C_3, \dots\}$, and corresponding first intermediate destinations $\mathcal{M} = \{m_1, m_2, m_3, \dots\}$. In particular, we call such holes $C_i, i > 2$ as *extra holes* given source s and destination t .

We will prove that the heights of sequentially ordered interfering holes in set \mathcal{C} monotonically decrease, at least, at the rate of $\frac{l}{\sqrt{h(C_1)^2 + l^2}} \cdot h(C_1)$, where $l = d(s, t')$. More formally, we will first show that, given two interfering holes $C_i, C_j \in \mathcal{C}$, if $i > j$, then $h(C_i) > h(C_j)$, and then we will prove the decrease-rate.

Without loss of generality, we consider only the case with interfering holes, C_1 and C_2 , defined for line segments \overline{st} and $\overline{sm_1}$, respectively. The cases for $C_j, j > 3$ can be reasoned in the same way. Consider a point t' (see Figure 17) that is an intersection between convex hull C_1 and line segment \overline{st} such that Euclidean distance to s is smallest. We denote $d(t', \overline{sm_1})$ by α for simplicity. Now consider an interfering hole C_2 that interferes line segment $\overline{sm_1}$. If we can show that $h(C_2) \leq \alpha$, then $h(C_2) \leq \alpha < h(C_1)$, because $h(C_1)$ is a hypotenuse of a triangle $\Delta t'm_1p$.

Assume in contradiction that $h(C_2) > \alpha$. Then, there are two cases to consider.

Case 1: the intermediate destination m_2 is selected in the *lower region* of line segment $\overline{sm_1}$ – specifically, when a line connecting s and m_1 bisects the two dimensional space into two areas, the lower region refers to the area containing t' . In this case, hole C_2 intersects with line segment \overline{st} , because $h(C_2) > \alpha$. This is a contraction, because C_2 must then be an interfering hole for \overline{st} .

Case 2: the intermediate destination m_2 is selected in the *upper region* of line segment $\overline{sm_1}$ – specifically, when a line connecting s and m_1 bisects the two dimensional space into two areas, the upper region refers to the area not containing t' . In this case, there always exists an intermediate destination m'_2 in the *lower region* such that $d(s, m'_2) + |m'_2 \sim m_1| < d(s, m_2) + |m_2 \sim m_1|$, a contradiction.

Therefore, we can conclude that $h(C_2) \leq \alpha$, so that $h(C_2) < h(C_1)$. In an inductive manner, we can find that $h(C_i) < h(C_j), i < j, \forall i, j$. Now we obtain the decrease rate. Consider triangle $\Delta st'm_1$, and denote the angle $\angle t'm_1p$ by β . We can then derive the following equation:

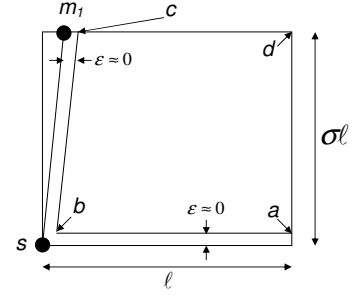


Fig. 18. Illustration for the scenario that maximizes the stretch.

$$\frac{\alpha}{h(C_1)} = \sin \beta = \frac{l}{\sqrt{h(C_1)^2 + l^2}}. \quad (1)$$

Thus, α is given as the following:

$$\alpha = \frac{l}{\sqrt{h(C_1)^2 + l^2}} \cdot h(C_1). \quad (2)$$

For example, if $h(C_1) = \frac{l}{2}$, the decrease-rate is $\frac{1}{\sqrt{5}}$; that is, the height of n -th interfering hole can be at most: $h(C_n) = h(C_1) \cdot (\frac{1}{\sqrt{5}})^n$. \square

B. Proof of Lemma 2

We consider a network of rectangular shape with sides l and σl , where $\sigma \geq 1$ as shown in Figure 18. Note that given any network, we can approximate the deployment area with a rectangular shape. Given the first interfering hole C_1 for \overline{st} , the outside-convex routing algorithm finds a sequence of *extra holes* $\mathcal{C} = \{C_2, C_3, \dots\}$, where, for each C_i , the first intermediate destination is denoted by m_i , as shown in Figure 17. Furthermore, each time the outside-convex routing algorithm discovers the i -th extra hole C_i , it can easily compute the length of the path P connecting s and m_1 as the following: $|P| = d(s, m_i) + d(m_i, m_{i-1}) + \dots + d(m_2, m_1)$. Obviously, if $|P| < d(s, t') + |t' \sim m_1|$, the algorithm proceeds with its next step; otherwise, the algorithm selects the path $s \sim t' \sim m_1$. This implies that $|s \sim m_1|$ can be at most $d(s, t') + |t' \sim m_1|$.

In particular, consider Figure 18 which depicts the case when the stretch of path $s \sim m_1$ is maximized, because the convex hull with maximum extent is placed so that $d(s, t') + |t' \sim m_1|$ is maximized. This case then yields the following:

$$d(s, m_1) \approx \sigma l \text{ and } d(s, t') + |t' \sim m_1| \approx (2 + \sigma)l. \quad (3)$$

And then, the stretch of path $s \sim m_1$ is naturally given as the following:

$$\frac{(2 + \sigma)l}{\sigma l} = \left(1 + \frac{2}{\sigma}\right). \quad (4)$$

Using Equation 4, we prove that, given any source s and destination t , the stretch of path $s \sim t$ is a constant. Assume that the parameter h is given as a sufficiently large number

so that information on existing VON nodes is known to all nodes in a network (e.g., $h > \max_hop_count$). Thus, s can find all interfering holes for \overline{st} . Specifically, the outside-convex algorithm identifies the set of intermediate destinations denoted by $\{I_1, I_2, \dots, I_n\}$ on the interfering holes for \overline{st} such that the length of path P' connecting s and t , $|P'| = d(s, I_1) + d(I_1, I_2) + \dots + d(I_n, t)$ is minimized. We term each $d(x, y)$ for $|P'|$ as *path segment*. Now the last issue is the *extra holes* that influence the stretch of each path segment. Note that if we do not take *extra holes* into account, $|P'|$ is theoretically optimal. Equation 4 shows that the length of each path segment can be, however, increased by up to the factor of $(1 + \frac{2}{\sigma})$. Thus the length of path P'' that takes the extra holes into account can be increased as follows:

$$|P''| \leq (1 + \frac{2}{\sigma})|P'|. \quad (5)$$

The interpretations of Equation 5 are:

- for the network of pipe-line-shape (i.e., $\sigma \rightarrow \inf$), the path stretch converges to 1. The intuition behind this result is that as the network becomes narrow, the space for *extra holes* becomes smaller, thereby yielding optimal result of stretch 1.
- conversely, the performance of outside-convex routing is worst when the network is of square-shape. The reason is simple because such network allows the largest chance for sources to identify *extra holes*. Specifically, the path stretch for such network is given as 3.

□

C. Proof of Theorem 1

Assume that a network is given with deployed holes. For now, we do not consider holes that contain source s and destination t . We will take such holes into account later in this proof. Given a $s-t$ pair, the outside-convex routing identifies intermediate destinations $\{I_1, I_2, \dots, I_n\}$. Without considering convex hulls that contain s and t , the length of path P_1 connecting source s and destination t is given as follows:

$$|P_1| = |s \sim I_1| + \sum_{i=2}^{n-2} |I_i \sim I_{i+1}| + |I_n \sim t|. \quad (6)$$

Note that, assuming parameter h is set to a number that is greater than the maximum hop count of the network, so that the information on VON nodes are broadcast to all nodes in the network, path length $|P_1|$ is optimal without considering the two factors:

- extra interfering holes for the path segments – that is, *extra holes* for $(s \sim I_1)$, $(I_i \sim I_{i+1})$, and $(I_n \sim t)$.
- the holes that contain s and t .

We first consider extra interfering holes for the path segments. By Lemma 2, the length of path P_1 can be increased by up to a factor of $1 + \frac{2}{\sigma}$. We denote this new path with increased length by P_2 . And then, the length of path P_2 is given as follows:

$$|P_2| \leq (1 + \frac{2}{\sigma}) \cdot |P_1|. \quad (7)$$

Finally, if we take the holes containing s and t into account, the stretches of path segments $s \sim I_1$ and $I_n \sim t$ are influenced. Specifically, Lemma 4 describes that the stretch of such path segment is $\frac{\pi \cdot r'}{2}$, where r' is the maximum diameter of holes in the network. Therefore, the length of path P_2 that takes the holes containing s and t into account is given as the following. We denote this final path by P_3 .

$$|P_3| \leq \frac{\pi \cdot r'}{2} \cdot (1 + \frac{2}{\sigma}) \cdot |P_1|. \quad (8)$$

This proves the theorem. □