Virtual position based geographic routing for wireless sensor networks

Jiaxi You\textsuperscript{a,*}, Qi Han\textsuperscript{b}, Dominik Lieckfeldt\textsuperscript{a}, Jakob Salzmann\textsuperscript{a}, Dirk Timmermann\textsuperscript{a}

\textsuperscript{a}Institute of Applied Microelectronics and Computer Engineering, University of Rostock, Richard-Wagner-Str. 31, 18119 Rostock, Germany
\textsuperscript{b}Department of Mathematical and Computer Sciences, Colorado School of Mines, 1500 Illinois Street, Golden, CO 80401, USA

A R T I C L E   I N F O

Article history:
Available online 4 January 2010

Keywords:
Geographic routing
Greedy forwarding
Routing hole
Wireless sensor networks

A B S T R A C T

As sensor networks are deployed over various terrains, the complexity of their topology continues to grow. Voids in networks often cause existing geographic routing algorithms to fail. In this paper, we introduce a novel concept: virtual position to address this issue. A virtual position is the middle position of all direct neighbors of a node. Such virtual position reflects the neighborhood of a sensor node, as well as the tendency of further forwarding. Instead of comparing nodes' real geographic positions, virtual positions are compared when selecting the next hop. For sparsely-deployed networks, this technique increases success rate of packet routing without introducing significant overhead. We here present an algorithm using this foundation concept and then design several enhanced versions to improve success rate of packet routing in sensor networks. We also conduct complexity analysis of the algorithms and support our claims of the algorithms’ superiority with extensive simulation results.

© 2009 Elsevier B.V. All rights reserved.

1. Introduction

Wireless Sensor Networks (WSNs) are composed of many sensor nodes, which are capable of sensing physical parameters, processing data and communicating wirelessly [1]. Sensor nodes are usually battery powered and left unattended after deployment. Since radio communication among sensor nodes is the main drain of energy in WSNs, energy efficiency remains a critical design issue for WSNs.

Among various routing algorithms, on-demand routing algorithms such as AODV [2] are popular, while their flooding-based route discovery often lead to high control overhead [3]. In contrast, single-path geographic routing with Greedy Forwarding (GF) is attractive for WSNs [1]. In a basic GF algorithm, a node communicates only with its direct neighbors (1-hop). The neighboring node that further minimizes the remaining distance of a packet to its destination will be selected as the next hop. Such localized approach is effective and can be dynamically adapted to changes, which only require position information of sensor nodes [1].

As modern WSNs are becoming popular in various applications, their topology is becoming complicated. Due to limited precision of deployment, voids can cause routing holes in the network, which often lead traditional GF algorithms to fail [6]. The reason is the local minimum phenomenon illustrated in Fig. 1. In single-path GF routing algorithm, forwarding of packets towards the sink can fail at node A, since there is no direct neighbor closer to the destination than node A itself. Besides, uneven power consumption on sensor nodes can also lead to new routing holes in the later lifetime of WSNs.

The objective of our work is to improve the success rate of GF for sparse WSNs or WSNs with small routing holes. To this end, we introduce a novel concept “virtual position”, which is the middle position of all direct neighbors of a node. This virtual coordinate system provides an indication of how the neighbors are distributed around a node. We present a basic geographic routing algorithm using virtual position and several improved versions. All simply employ GF based on the virtual coordinates throughout the routing process. The basic assumption of our approach is that each node in the network is aware of its own geographic position as well as the positions of its direct neighbors.

Our approach distinguishes itself from existing geographic routing algorithms [7,8,13] with the following features. (1) Improved packet delivery rate. We prove that our approach achieves comparable performance to common hole-bypassing algorithms based on face routing, especially for sparse WSNs. The multi-path version of our approach tries to avoid holes by selecting multiple next hop nodes using both geographic and virtual positions of the node. Although our approach does not completely eliminate the local minimum problem, it tries to steer packets from local minimum in advance. (2) Low overhead. Our approach is strictly localized, and avoids the computation and maintenance of the planar graph. Therefore, the amount of control overhead of the proposed algorithms is strictly limited. Our approach is a one-phase routing process that does not involve the face routing [8] used in common hole-bypassing algorithms. (3) Short routing path. Our approach inherently results in high routing efficiency as the basic GF algorithms.
The performance of face routing in dynamic networks. In [14], the authors introduced Probabilistic Geographic Routing protocol (PGR) to use residual energy of nodes and link reliability as routing parameters. Based on these parameters, PGR assigns candidate nodes with the probability of being the next hop. Using nodes with more residual energy and more reliable links can increase delivery rate of packets, as well as lifetime of the network. For nodes in a highly mobile and noisy environment, the concept of lifetime timer [15] is applied to nodes in the network, which is used as the criterion of selecting the next hop.

Since GPSR only starts to detour packets around a hole when local minimums are met, the paths constructed by face routing are typically not the best path available to bypass the hole. Recently, few techniques [16,17] were introduced to construct optimal detour paths around routing holes. In [17], the authors proposed a geographic routing algorithm called Hole-BYpassing routing with Context-AwareNess (HobyCan). In HobyCan multiple detour paths are set up locally, which can be used alternatively to achieve optimal routing paths or load balance of the network. However, such approach only applies to the many-to-one communication model, where all packets are sent to the static base station.

In [10], Stojmenovic et al. proposed to use GF with information of 2-hop neighbors (2-hop GF). The success rate of the GF algorithm is improved with each node aware of its direct neighbors as well as its 2-hop neighbors (neighbors of its neighbors). Among all direct and 2-hop neighbors, packets are sent to the node which is the closest to the destination via 1-hop or 2-hop forwarding. Since farther neighbors are available for the selection of next hop, small routing holes can be avoided efficiently. The authors also proved that such GF based methods are inherently loop-free. The method in [10] solely employs the basic idea of GF during routing. Such a method limits control overhead since it only requires local knowledge of sensor nodes (information of 1-hop and 2-hop neighbors) and avoids sophisticated complementary phases to recover from local minimums. However, 2-hop GF is not strictly localized to the direct neighbors, therefore, it is not scalable for further extension. For example, when further improvement of success rate is demanded, nodes need to have information of neighbors within K hops (K ≥ 3). For a 2D wireless network, the number of neighbors stored on each sensor nodes increases proportionally to K^2. Similarly is the control overhead for the maintenance of neighbor information within K hops.

2.2. Routing based on virtual coordinate systems (VCS)

In two-phase based geographical routing, the performance during the GF phase is much better than during the hole-bypassing phase [9]. To increase the ratio of GF during routing, Liu et al. presented the idea of Aligned Virtual Coordinate System (AVCS) [9]. Virtual coordinates are based on integer number of hops to the reference nodes (anchor nodes) and represent a coarse approximation of node locations. The success rate of greedy routing is improved with AVCS, while flooding of control message from anchor nodes to the whole network causes large control overhead during the set up of AVCS.

Similar to Hop ID [18], each node must obtain a vector of minimum hop distances to some randomly selected landmark nodes in the network. Such multidimensional coordinates are stored on each node as its Hop ID. Packets are routed in a greedy forwarding manner based on the Hop IDs. Such scheme effectively avoids the routing holes, while building Hop ID on each node requires a considerable amount of global message communication and storage overhead.

3. Network model and problem statement

3.1. Network model

We consider a WSN deployed in a region Ω, which is typically a 2D square or disk. The network can be modeled by a communication graph G = (V,E), where V = {v1, ... , vn} is a set of N = |V| sensor nodes.
nodes and $E = \{e_1, \ldots, e_M\}$ is a set of $M = |E|$ undirected links. Nodes are randomly deployed and are assumed to be static after deployment. Every node has a uniform transmission range $r$. We used a simple disc-communication model: sensor nodes in the transmission range can receive signals from a transmitter without loss. Each node is aware of its own position and the position of its 1-hop neighbors. For any pair of nodes, a link node is aware of its own position and the position of its 1-hop neighbors. Assume node $v$ has a set of $n$ direct neighbors: $V_v = \{V_{v,1}(x_{v,1}, y_{v,1}), V_{v,2}(x_{v,2}, y_{v,2}), \ldots, V_{v,n}(x_{v,n}, y_{v,n})\}$, the virtual position of node $v$ is:

$$\langle x_A, y_A \rangle = \left( \frac{1}{n} \sum_{i=1}^{n} x_{v,i}, \frac{1}{n} \sum_{i=1}^{n} y_{v,i} \right)$$

(1)

3.2. Problem statement

In the common many-to-many routing scenario, a packet appears on an arbitrary node (the source node $s$) and is routed to another arbitrary node (the destination node $d$) in the network. Upon receiving a packet, a node forwards the packet to the neighboring node $v$ according to a specific routing algorithm. Given a routing algorithm $A$, a path is denoted as the set of individual forwarding steps $P_A = \{V_{A,1}, \ldots, V_{A,l}\}$ as a subset of $E$, where $l$ is the hop count of the routing path. A packet with the source and destination pair $S, D$ is considered to be delivered when at least one path is found by $A$ connecting $S$ and $D$. Algorithm $A$ is called "localized" if only the information of the current node and the information of its K-hop neighborhood is required. Here $K$ is usually a constant of 1 or 2. The objective of this paper is to improve localized geographical routing algorithms in terms of packet delivery success rate for sparse WSNs or WSNs with small routing holes.

4. Virtual position based greedy forwarding

Our approach revolves around a novel concept-virtual position. The virtual position of a node is the middle point of all its direct neighbors. Assume node $A$ has a set of $n$ direct neighbors: $V_A = \{V_{A,1}(x_{A,1}, y_{A,1}), V_{A,2}(x_{A,2}, y_{A,2}), \ldots, V_{A,n}(x_{A,n}, y_{A,n})\}$, the virtual position of node $A$ is:

$$\langle x_A, y_A \rangle = \left( \frac{1}{n} \sum_{i=1}^{n} x_{A,i}, \frac{1}{n} \sum_{i=1}^{n} y_{A,i} \right)$$

(1)

Each node calculates its virtual position according to Eq. (1), and broadcasts its virtual position to its direct neighbors. The information of virtual position is stored on nodes themselves and their direct neighbors. In other words, each node has the knowledge of its own virtual position, and the virtual positions of its direct neighbors. Fig. 2(a) shows the geographic positions of nodes. Fig. 2(b) shows the calculated virtual positions of nodes. The virtual position of node 11 is the same as its real geographic position, since its direct neighbors (node 6, 7, 8, 10, 12, 14, 15, 16) are located uniformly in the transmission circle centered at the geographic position of node 11. In contrast, the virtual position of node 12 (Fig. 2(b)) locates to the left of the geographic position of node 12 (Fig. 2(a)), because the positions of its direct neighbors (node 7, 8, 11, 15, 16) are left-biased in the transmission circle of node 12.

In the following, we describe our geographic routing ideas in detail. Routing algorithm “Greedy Forwarding with Virtual Position
(VIP)” is the basic approach using the idea “virtual position”. We also propose two improved versions, the “Greedy Forwarding with Hierarchical Virtual Position (HVP)” algorithm and the “Restricted Multipath Greedy Forwarding with Virtual Position (RM-VIP)” algorithm to further improve the performance in terms of packet delivery.

4.1. Basic greedy forwarding with virtual position (VIP)

We propose “Greedy Forwarding with Virtual Position (VIP)”, a look-ahead geographic routing algorithm based on the coordinate system of virtual positions. Instead of using farther neighbors as in 2-hop GF, VIP uses the virtual positions of nodes to involve farther neighbors in the look-ahead routing process. Our algorithm is strictly localized, where a node only have the information of its direct neighbors.

In traditional GF routing, Greedy algorithm[4] and the Most Forwarding progress within Radius (MFR) algorithm[5] are two fundamental versions of GF based on distance. In other words, they both require the next hop to be closer to the destination than the current node. However, Greedy selects the neighbor that is the closest to the destination; and MFR selects the neighbor that makes the most progress along the direction of packet forwarding. Fig. 3 illustrates the slight differences between the Greedy algorithm and the MFR algorithm.

VIP has the same two variants called Greedy-VIP and MFR-VIP, which are based on the same principle of the Greedy and MFR algorithm, respectively.

- **Greedy-VIP**: As an example for a 2D WSN, when a packet with destination $D(x_0, y_0)$ arrives at node $A(x_A, y_A)$, Greedy-VIP evaluates its own virtual position $(x_A, y_A)$ and the virtual positions of its direct neighbors. The neighbor with the virtual position that is the closest to node $D$ is selected as the next hop. According to the Greedy algorithm, the virtual position of the selected neighbor must also be closer to the destination than the virtual position of current node $A$. Namely, the next hop node $v_{A_i}$ of Greedy-VIP meets the following two conditions:

\[
\arg\min_{A_i} \left( \sqrt{(x_{A_i} - x_0)^2 + (y_{A_i} - y_0)^2} \right) 
\]

and:

\[
\sqrt{(x_{A_i} - x_0)^2 + (y_{A_i} - y_0)^2} < \sqrt{(x_A - x_0)^2 + (y_A - y_0)^2}
\]  

- **MFR-VIP**: It selects the neighbor that makes the greatest progress towards the destination of a packet. Given the vectors $DA$ and $DV_{A_i}$, the next hop node $V_{A_i}$ is the one that has the minimal dot product of the two vectors:

\[
\arg\min_{A_i} (DA \cdot DV_{A_i})
\]

In addition, $V_{A_i}$ also satisfies Eq. (3).

VIP fails when there is no neighbor that has a virtual position to make further progress towards the destination of a packet. Without mentioning specifically, each enhanced version also has these two variants (based on the principle of Greedy and MFR, respectively.), and we will mostly use Greedy to illustrate the ideas.

The virtual position of a node provides an indication of how the direct neighbors are averagely located around the node, hence it is a suitable metric to demonstrate the tendency of further forwarding during geographic routing. Consider the example in Fig. 2, for a packet that needs to be sent from node 10 to node 13, both Greedy and MFR using the traditional GF get stuck at node 12; since there is no neighbor that can make further progress towards the destination. In contrast, the virtual position of node 12 is strongly left-biased due to the void on the right side of node 12. As a result, MFR-VIP finds a path (10→11→12→16→21→17→13) around the hole using virtual positions of nodes.

With this idea, nodes only need to interact with their direct neighbors to obtain information of their geographic positions and virtual positions. Therefore, setting up virtual positions is strictly localized. For a static network, only 2 rounds of information exchange between 1-hop neighbors are needed during setup. In the first round, each node transmits its geographic position to all direct neighbors. The virtual position of a node is calculated after the first round. In the second round, each node transmits its calculated virtual position to all direct neighbors. The message overhead in the network is $O(n)$, where $n$ is the total number of nodes.

To further improve the success rate of GF, we generalize VIP to “higher level virtual position” that considers farther nodes (neighbors of K-Hop, $K \geq 1$). If we refer to the virtual position derived using Eq. (1) as the st-level virtual position and the geographic position as the 0th-level virtual position, then the kth-level virtual position of node A can be computed from the $(K - 1)$th-level virtual positions of $K$ neighbors and its direct neighbors. Assume node A has a set of $n$ direct neighbors: $VA=\{V_{A_1}(x_{A_1},y_{A_1}), V_{A_2}(x_{A_2},y_{A_2}), \ldots, V_{A_n}(x_{A_n},y_{A_n})\}$, the Kth-level virtual position of A is calculated as follows.

\[
(x_k^A, y_k^A) = \left( \frac{x_{A_1}^\star + x_{A_2}^\star + \cdots + x_{A_n}^\star}{n}, \frac{y_{A_1}^\star + y_{A_2}^\star + \cdots + y_{A_n}^\star}{n} \right)
\]

Such Kth-level virtual positions of nodes indicate how the K-hop neighbors are located on average, also the tendency of further forwarding during geographic routing. For $K \geq 1$, $(K+1)$ rounds of information exchange between 1-hop neighbors are needed during setup. We will use VIP($K$) to represent this family of algorithm, where $K \geq 1$. Fig. 2(c) is an example of VIP using 2nd-level virtual position, which demonstrates that using virtual position of higher level indicates better forwarding tendency in geographic routing, since it takes farther neighbors into consideration. VIP($K$) fails when there is no neighbor that has the Kth-level virtual position making further progress towards the destination of a packet.

4.2. Greedy forwarding with hierarchical virtual position (HVP)

VIP($K$) can introduce new local minimums during routing. For instance, in Fig. 4, a packet needs to be sent from node 5 to node 4. When using the geographic positions of nodes, the packet can be successfully delivered (path 5→3→4 in Fig. 4(a)). While using 1st-level virtual position, node 5 has no neighbor to make further progress towards node 4, thus becomes a local minimum.

To address this problem, we introduce the “Greedy Forwarding with Hierarchical Virtual Position (HVP)” algorithm (Table 1 shows...
the pseudo code of HVP, HVP(K) uses the combination of all K-level virtual positions (K ≥ 1) and the geographic positions of nodes in a down-hill fashion. Namely, when a local minimum is met using K′th-level virtual position, HVP(K) will lower the level of virtual position from K′th-level to (K − 1)th-level (K ≥ 2), or from 1st-level virtual position to geographic position, HVP requires nodes to store the geographic positions as well as all the K-level virtual positions of itself and its direct neighbors. A flag flag_level is added to the packets to indicate the current level of virtual position (using geographic position when K = 0). In our example (Fig. 4(b)), the packet stuck (using 1st-level virtual position) at node 5 can set its flag_level to be 0, and reach its destination via the path 5–3–2–1 by using the geographic positions of nodes.

To ensure that HVP is loop-free, such down-hill process is unidirectional. Namely, for a packet initialized with Kth-level virtual position and adjusted to (K − 1)th-level virtual position after meeting a local minimum, the packet stays on (K − 1)th-level and can only be further adjusted to the lower levels (or from 1st-level virtual position to geographic position). HVP fails if the lowest level virtual position (the geographic position) is already used and there is no neighbor to make further progress towards the destination of a packet.

Note that the down-hill process does not need to have a fixed decrement of 1. When using larger decrements, less levels of virtual position are needed, this implies less information storage on nodes. For instance, the flag_level can be decreased by 2 instead of 1, every time a local minimum is encountered. Dynamic decrements can be used when certain level of virtual position is missing.

### 4.3. Restricted Multipath Greedy Forwarding with Virtual Position (RM-ViP)

ViP and HVP only unicast a packet to the next hop based on a certain level of virtual position. At each node, using different levels of virtual position may result in different next hop selections. To further reduce the possibility of hitting a local minimum, we may use GF in a way that each node compares with its neighbors using all available levels of virtual position, and multicasts the packet to all next hops calculated. For instance, during packet routing on the Kth-level, up to (K + 1) different next hops may be calculated with all Kth-level virtual position and geographic position. However, such multicasting tends to have high computational and communication overhead. During multicasting, the number of next hops needs to be limited.

Based on this idea, we propose the Restricted Multipath Greedy Forwarding with Virtual Position (RM-ViP) algorithm (Table 2 shows the pseudo code of RM-ViP). RM-ViP is a practical multipath geographic routing algorithm extended from ViP. In RM-ViP(K), only a specific Kth-level (K ≥ 1) and the geographic position are employed. When a packet arrives, a node selects the next hops based on the Kth-level (K ≥ 1) virtual position and the geographic position, respectively. If the two selections are identical, the node unicasts the packet to the next hop as in ViP. Otherwise, the routing path forks and two copies of the packet are sent to the different next hops. A node sends at most two copies of a packet to limit routing overhead. In RM-ViP, copies of the same packet may take different routes and, hence, arrive at a node in different time. To avoid sending redundant packets, each node records the packet ID for a certain time Tmult after forwarding the first copy of the packet. During Tmult, any arriving packet with the same ID will be dropped. That way, each node forwards the same packet only once, which further reduces the number of packets sent. An appropriate Tmult also avoids memory overflow due to recording packet IDs.

Fig. 5 shows an example of Greedy-RM-ViP where K = 2. The packet is routed from node 10 toward node 13. Fig. 5(a) shows the geographic positions of nodes. Fig. 5(b) shows first step of packet forwarding, where the routing path forks on node 10. Node 15 is selected as the next hop of node 10, based on their 2nd-level virtual positions. Node 11 is also selected as the next hop of node 10, based on their geographic positions. Node 10 multicasts the packet both to node 15 and node 11. In Fig. 5(c), node 11 multicasts to node 16 and node 12, according to their 2nd-level virtual positions and geographic positions, respectively. In the same way, node 15 multicasts to node 20 and node 12. Packets arrived at node 12 will be consumed, since node 12 can not find a next hop neither using 2nd-level virtual position nor geographic position. In Fig. 5(d), node 16 continues to multicast to node 21 and 12. Here node 20 only sends the packet to node 21, since node 21 is selected both by 2nd-level virtual position and geographic position. Because redundant packets arriving at the same node are dropped, only one copy of the packet is sent from node 21. During further forwarding,
each node continues to perform RM-ViP in every step. Since the selections based on the 2nd-level virtual position and geographic position are identical in the following steps, the packet is routed to the destination in a single-path. RM-ViP(2) fails when there is no neighbor that has the 2nd-level virtual position or geographic position to make further progress towards the destination for any copy of a packet.

5. Algorithm properties

In this section, we will show that the proposed ViP, HVP and RM-ViP algorithms are strictly localized, and can terminate based on local knowledge, i.e., they are loop-free. We will also present theoretical analysis of the algorithms’ complexity in terms of communication, storage, and computation.

Lemma 1. For any static network, ViP is loop-free.

Proof. As proved in [10], single-path routing algorithms based on the Greedy algorithm or the MFR algorithm are inherently loop-free. ViP employs the virtual position coordination system which is the 1-to-1 mapping of the original geographic coordinate system of a network. Instead of the geographic positions, ViP evaluates the corresponding virtual positions of nodes based on the coordinate system of virtual position. Thus, Greedy-ViP and MFR-ViP are also inherently loop-free. As a consequence, ViP is loop-free.

Assume that ViP of Kth-level virtual position \( K \geq 1 \) is loop-free. Since ViP of \((K + 1)\)th-level virtual position employs the 1-to-1 mapping of the Kth-level virtual position coordinate system. Therefore, ViP of \((K + 1)\)th-level virtual position is loop-free. As proved above, ViP of 1st-level virtual position is loop-free. Thus, ViP of Kth-level virtual position is loop-free for any \( K \geq 1 \).}

Lemma 2. For any static network, HVP is loop-free.

Proof. HVP starts with Kth-level virtual position \( K \geq 1 \) and decreases the flag level of packets until 0, which refers to the coordinate system of geographic position. As proved in Lemma 1, ViP of Kth-level virtual position is loop-free. Therefore, each step in the down-hill scaling of HVP is loop-free. Thus, HVP starts with Kth-level virtual position is loop-free for any \( K \geq 1 \).

Lemma 3. For any static network, RM-ViP is loop-free.
Proof. The multiple paths in RM-ViP can be considered as the composition of individual ViP paths starting from involved nodes, based on either Kth-level virtual position or geographic position. As proved in Lemma 1, ViP of Kth-level virtual position is loop-free for any K \geq 1. Therefore, RM-ViP is loop-free. □

Algorithms’ complexity during setup and execution phases:

- Communication overhead during algorithm setup phase: As mentioned earlier, the information of virtual position of nodes and their neighbors is calculated only with local information. While a higher level virtual position is derived from the virtual position of 1 level lower. The message overhead during the setup of the virtual position is shown in Table 3. GF and GPSR need 1 round of information exchange between neighbors. We refer GF using K-hop neighbor information as GF(K), K \geq 1. GF(K) needs K rounds. For the proposed algorithms based on virtual position, (K + 1) rounds of iterations are performed to acquire the Kth-level virtual position of a node and its 1-hop neighbors. We assume a static network where topology is static, namely position and extend of voids in the network does not change after deployment.

- Communication overhead during algorithm execution phase: In a GF algorithm, a node needs to evaluate all the entries of neighbors stored on the node, and selects the next hop based on a certain criterion. Therefore, the average storage and computational overhead is proportional to the number of neighbor entries on nodes. This metric also reflects the control message overhead due to maintaining the information of virtual positions. Table 4 shows the storage and computational overhead of related algorithms. GF or ViP only require either the geographic position or a Kth-level virtual position of a node itself and its 1-hop neighbors. GPSR stores an extra 1-to-1 planar graph derived from the original network model. When using 2-hop GF, all the neighbor information within 2 hops is stored. Since PVP(K) starts with Kth-level virtual position, all K levels of virtual position and the geographic position of a node and its direct neighbors are needed. In contrast, RM-ViP only requires the Kth-level virtual position and geographic position. Here we assume that nodes are deployed in a 2D network with the same density.

### 6. Performance evaluation

We implemented the proposed algorithms using a probabilistic sensor network simulator Prowler [19]. Prowler is a probabilistic wireless network simulator providing realistic radio/MAC models for the Berkeley mote platform. We used the default probabilistic radio model in Prowler that determines the strength of a transmitted signal at a particular distance from the transmitters. The transmission model is given by:

\[
P_{\text{rec,ideal}}(d) = P_{\text{transmit}}/(1 + d^\gamma), \quad \text{where } \gamma = 2
\]

\[
P_{\text{rec}} = P_{\text{rec,ideal}} \times (1 + \zeta(x)) \times (1 + \beta(t))
\]

where \(P_{\text{rec,ideal}}\) is the ideal received signal strength at distance \(d\) without noise, \(\zeta\) and \(\beta\) are random variables with normal distributions \(N(0, \sigma_\zeta)\) and \(N(0, \sigma_\beta)\), respectively. \(P_{\text{transmit}}\) is the signal strength at the transmitter. In this model, \(\zeta\) depends only on the distance between node \(i\) and node \(j\), and \(\beta\) changes over time. Furthermore, an additional parameter \(P_{\text{error}}\) models the probability of a transmission error by all other sources/effects. A signal is received if the signal strength at the receiver is greater than a reception threshold \(\text{limit}\), which is set to be 0.1 as default. The MAC-layer model is a basic Berkeley motes’ CSMA MAC protocol. There is a collision of packets if more than one transmission arrives at the same time at a node.

![Fig. 6](image)

**Fig. 6.** Impact of node density on average success rate, \(K = 1, \ldots, 5\).
The simulated network is a 50m × 50m square plane, where sensor nodes are randomly deployed. Packets are generated with random pairs of source-destination addresses. Different numbers of sensors are tested in the simulated area. Packets are generated with random pairs of source-destination addresses. Different numbers of sensors are tested in the simulated area. We used the default value \( \sigma = 0.45 \) in Prowler to simulate the distance dependence of radio channel. With transmission power of 1 unit, the ideal transmission range of sensor nodes is 3 m. We used the default probabilistic radio model in Prowler that determines the strength of a transmitted signal at a particular distance from the transmitters. To capture the performance with random signal loss over time, we show both results using the probability of a transmission error \( P_{\text{error}} = 0.05 \) and \( P_{\text{error}} = 0 \).

To capture the generality of random WSN deployment, we did not manually add void to the network. Therefore, we use the density of deployment to control and change the probability of encountering routing holes in our simulations. For the same deployment area, lower density implies sparser layout of a network and bigger chance to encounter routing holes during packet forwarding.

The objective of our experiments is to demonstrate the performance of our algorithm in sparsely-deployed WSNs, or WSNs with small voids due to non-uniform deployments. We show only the results of the variant Greedy-ViP, Greedy-HVP and Greedy-RM-ViP in detail. The performance of MFR-ViP, MFR-HVP and MFR-RM-ViP are similar from our simulation. The results were compared with the GPSR algorithm (based on RNG planar graph).

We use the following metrics to compare the performance of the simulated routing algorithms:

- **Success rate**: This is defined as the percentage of successful packet delivery. Packets with random source-destination pairs were generated and forwarded in the simulated network. When the density of deployment is low, voids appear frequently in the network. This metric illustrates the ability of avoiding routing hole of the simulated algorithms.
- **Delay**: This is defined as the average hop count of packets from their source to the destination. Given the same success rate, small packet delay (in hops) represents better performance in terms of routing efficiency and energy consumption.
- The number of neighbor entries stored on sensor nodes implies the storage overhead, as well as the computational overhead due to selecting the next hop. Furthermore, the control message overhead of maintaining neighbor information is also proportional to the number of entries.

---

![Graph 1](image1.png)  
**Fig. 7.** Impact of node density on average success rate, \( K = 1, P_{\text{error}} = 0 \).

![Graph 2](image2.png)  
**Fig. 8.** Impact of node density on average success rate, \( K = 5, P_{\text{error}} = 0 \).
6.1. Experimental results

Fig. 6 shows the average percentage of successful packet forwarding from our simulation. Fig. 6(a) compares the performance of the Greedy algorithm and Greedy-ViP using $K$th-level virtual position (Greedy-ViP($K$), $K = 1 \ldots 5$) using $P_{\text{error}} = 0$. When the number of sensor nodes is small, nodes have relatively few neighbors. This leads to low success rates of the simulated algorithms. Packet routing fails either the source and destination nodes are not connected in the connectivity map, or there occurs a local minimum during forwarding. All hole-bypassing routing algorithms only deal with the failure due to the local minimum problem. Compared to Greedy, Greedy-ViP(1) shows a significant improvement of the success rate, which demonstrates its ability to avoid routing holes. As the level of virtual position increases, the success rate of Greedy-ViP improves, since farther neighbors are considered with virtual position of higher levels. All the simulated algorithms reach high delivery of packets in densely-deployed networks. In Fig. 6(a), when further increase the level $K$ beyond a certain number (e.g. when $K > 3$), the improvement of success rate becomes unconspicuous and tends to be upper-bounded. This is because in the calculation of virtual position of higher levels, the impact of farther neighbors is becoming minor during the iterations. Therefore, the tradeoff between the improvement and cost of using higher levels of virtual position can be exploited for specific networks. For a random deployed network as in our simulation, we consider $K = 5$ as the optimal point of such a tradeoff. In Fig. 6(b), the overall success rate is decreased due to random packet loss over time.

Fig. 7(a) illustrates the performance of using the 1st-level virtual position, in comparison to Greedy and GPSR. Compared to the corresponding Greedy-ViP algorithm, the down-hill scaling in the Greedy-HVP algorithm starts with the same level of virtual position has higher success rate due to its additional search on virtual positions of lower levels and the geographic positions of nodes when comes to local minimums. Greedy-RM-ViP shows the best performance among the 3 variants using the 1st-level virtual position, since it explores the full possibility of forwarding using both 1st-level virtual position and geographic position, in each step starting right from the source nodes. As in Fig. 8(a), Greedy-ViP(5) results in even better success rate since it considers 5-hop neighbors. In a random network without intentionally
placed holes, the success rate of our algorithm is comparable to GPSR.

In Fig. 7(b) and Fig. 8(b) with random signal loss over time, our algorithms gain further advantage over GPSR. The reason is our algorithms solely use the Greedy Forwarding throughout the routing process, and start detours around routing holes in advance. Compared to GPSR, therefore, our algorithms inherently result in smaller delay in terms of hop count (as shown later in the results). In realistic radio communication with probabilistic signal loss, routing algorithms with fewer hop counts have certain advantage subjecting to packet loss during forwarding. The success rate of RM-ViP is dramatically improved with random packet loss, because when a packet is lost on a node, its multiple copies will probably arrive later and make up to the loss.

Fig. 9 shows the average delay of the successfully delivered packets from our simulation versus varying node density in the same way as in Fig. 6. The hop count of routing paths decreases with increasing density, due to the enriching choice of nodes to make a straight route. In Fig. 9(a), algorithms with higher success rate have bigger average delay, since packets can be delivered over longer distances with these algorithms. As mentioned earlier, GPSR only starts to bypass a hole when a local minimum is met, which may lead to inefficient routing paths. The proposed look-ahead routing algorithms consider the tendency of future forwarding using virtual position, which allows the detour process to start in advance. In Figs. 10(a) and 11(a), when the density is low, the proposed single-path algorithms (ViP and HVP) show better delay performance than GPSR in the presence of routing holes. For each packet, RM-ViP sums the delay of the multiple copies of the same packet. Therefore, the delay factor of RM-ViP is much higher than the others. The hop counts in Fig. 9(b), Fig. 10(b) and Fig. 11(b) are smaller than their counterparts, because fewer packets are delivered due to random packet loss.

Fig. 12 illustrates the control overhead of the simulated algorithms indicated by the number of neighbor entries on nodes. In GPSR, nodes store the extra links to their neighbors of a planar graph. With improved performance in terms of success rate, ViP on a specific level has the same number of neighbor entries as in Greedy. This results in low storage and computational overhead of ViP. The number of neighbor entries of our Greedy-RM-ViP is approximately twice the entries of the Greedy algorithm. The storage and computational overhead of Greedy-HVP increases linearly along with the level of virtual position it starts with.

7. Conclusion

In this paper, we present a novel geographic routing algorithm named “Greedy Forwarding with Virtual Position (ViP)”, as well as its two extensions called the “Greedy Forwarding with Hierarchical Virtual Position (HVP)” algorithm and the “Restricted Multipath Greedy Forwarding with Virtual Position (RM-ViP)” algorithm. Each of the three algorithms has two variants based on the principle of the Greedy algorithm and the “Most Forwarding progress within Radius (MFR)” algorithm, respectively. Simulation results demonstrate that our proposed algorithms improve the success rate of geographic routing especially for sparsely-deployed WSNs and WSNs with small routing holes. The proposed algorithms solely employ Greedy Forwarding (GF) throughout the routing processes, and inherently result in high routing efficiency as the basic GF algorithms. Furthermore, the control overhead of our proposed algorithms is strictly limited.
Acknowledgements

This work is partially financed by the German Research Foundation (DFG), post graduate program MuSAMA (Multimodal Smart Appliance Ensembles for Mobile Applications).

References