

1-1-2012

## **A position based routing algorithm in 3D sensor networks**

Mohamed K. Watfa

*University of Wollongong, [mwatfa@uow.edu.au](mailto:mwatfa@uow.edu.au)*

Alaa M. Al Tahan

*American University of Beirut*

Follow this and additional works at: <https://ro.uow.edu.au/dubaipapers>

---

### **Recommended Citation**

Watfa, Mohamed K. and Al Tahan, Alaa M.: A position based routing algorithm in 3D sensor networks  
2012, 33-52.

<https://ro.uow.edu.au/dubaipapers/141>

# A Position Based Routing Algorithm in 3D Sensor Networks

Mohamed K. Watfa and Alaa M. Al Tahan

**Abstract**— As large-scale sensor networks become more feasible, properties such as stateless nature and low maintenance overhead make position based routing increasingly more attractive. Motivated by the fact that sensor networks would probably be deployed in a three dimensional space, we present a novel 3D geographical routing algorithm (3DGR) that makes use of the position information to route packets from sources to destinations with high path quality and reliability. The locality and high scalability of this algorithm make it suitable for wireless sensor networks. It provides high adaptability to changes in topology and recovery of link failures which increases its reliability. We also incorporate battery-aware energy efficient schemes to increase the overall lifetime of the network. To reduce latency, a method of keeping a small record of recent paths is used. We also show that location errors will still result in good performance of our algorithm while the same assumptions might yield bad performance or even complete failures in other popular geographical routing algorithms. We evaluate the 3DGR protocol using simulation. Compared to other geographic routing algorithms, we find that 3DGR exhibits noticeably longer network lifetime, smaller path stretch, smaller end-end delay and better packet delivery ratio.

**Index Terms**— 3D geographical routing, Sensor Networks, Location errors, Battery aware

## 1 INTRODUCTION

The use of wireless sensor networks forms a major part in next generation technology. Characteristics of sensor nodes make them suitable for use in many different fields like intrusion detection, environmental monitoring, and military applications. However, characteristics of sensor nodes require the design of new protocols that take into consideration resources scarcity in sensor nodes like memory and computing power. Another essential side that should be taken into consideration while designing a protocol for wireless sensor nodes is power consumption. Since sensor nodes are battery powered, energy becomes a limiting factor. In most cases, changing or recharging the battery might cost more than deploying a new node. Hence, extending the network lifetime is a critical metric in the evaluation of wireless sensor network protocols.

These factors make traditional routing algorithms like distance vector and link state not suitable for the use in wireless sensor networks. In an attempt to overcome these issues, new routing algorithms have been proposed using different approaches like greedy forwarding and geographical routing [1 - 6]. These new approaches handle sensor nodes restrictions by using local information about neighbor nodes. However, they have their own problems as summarized in Table I and they make their own assumptions which limit the use of such algorithms to specific environments that satisfy these assumptions. One of the major assumptions made by geographical al-

gorithms is assuming that nodes are deployed in a 2D plane. Such an assumption is invalid in real life scenarios and hence these algorithms cannot be applied in most situations. Three-dimensional modeling of the sensor network would reflect more accurately the real-life situations. Some applications of the results presented in this paper are:

- 1- *Disaster Recovery*: Natural disasters (floods, hurricanes, and fires) require sensing in different planes and thus 3-dimensional routing techniques are required. Three-dimensional networks also arise in building networks where nodes are located on different floors.
- 2- *Mapping Topographical Properties*: Random dense sensor deployment on irregular terrains like mountains and hills leaves the nodes lying on three dimensional surfaces that indicate the topographical properties of the terrain. Understanding the topography of an area enables the understanding of watershed boundaries, drainage characteristics, water movement, impacts on water quality, and soil conservation.
- 3- *Space Exploration* [7 and 8]: Wireless sensor networks will play an important role in planetary explorations. A rover functioning as a base station collects measurements and relays aggregated results to an orbiter.
- 4- *Undersea Monitoring* [9]: Underwater sensor deployment enables the real time monitoring of selected ocean areas. Under Water Acoustic Sensor Networks (UW-ASN) can consist of a number of sensors and submersible vehicles that are deployed to perform collaborative monitoring tasks over a given area.

• A.M. Al Tahan is with the Computer Science Department, American University of Beirut, E-mail: ama66@aub.edu.lb.  
 • M.K. Watfa is with the Computer Science Department, University of Wollongong, Dubai, E-mail: Mohamed.watfa11@gmail.com

Also, most geographic routing algorithms for sensor networks that were proposed in the last years were evaluated using simulation tools that were based on exact location information of each node. Since this is an unrealistic assumption in most sensor networks, the simulation results cannot be directly applied to real deployments. These unrealistic assumptions make the need of routing algorithms that work in three dimensional spaces a necessity to fit real applications.

In this paper, we present a new routing algorithm (3DGR) that is designed to work in real environments where nodes are distributed in a three dimensional space. 3DGR achieves the following desirable properties:

1. **Proactive Void Problem Anticipation:** Most geographical routing algorithms forward packets greedily to reach a destination. The power of greedy forwarding to route using only neighbor nodes' positions comes with one attendant drawback. There are topologies in which the only route to a destination requires a packet to move temporarily farther in geometric distance from the destination. This problem is known as the void problem. When the void problem occurs, geographical routing algorithms try to solve it using special techniques which cost more energy and time than if a different forwarding decision was made at a previous step. Also, some algorithms need to build their own special structures for the network like planarizing the network or building a routing tree. On the other hand, the novelty in our contribution is that unlike other protocols that react upon the detection of a void region, our protocol anticipates the occurrence of a void problem and tries to avoid it. 3DGR does not build any special structure in order to route packets. It tries to anticipate the occurrence of the void case and tries to avoid it before it happens. Although this is done by using information about one hop neighbors, the sender will be able to know information about two hop neighbors without any additional message exchange. This is done by simple aggregation and distribution of the work over neighboring nodes and making each of them use its one hop neighbors' information to send back to the sender. This is done at no additional cost since each node already has information about its one hop neighbors.
2. **Backtracking Technique:** When a void problem occurs (two hops void exists and this has much lower frequency than one hop void) a backtracking technique is used.
3. **3D Geocasting Technique:** To reduce the number of nodes involved in routing and hence save energy, we use a simple 3D geocasting algorithm to limit the one hop neighbors that would participate in the routing protocol to those that are in the right direction towards the destination.
4. **Energy and Bandwidth Efficiency:** Each request message is small compared to the control messages used by proactive protocols (that have to carry routing table) and to those used by reactive protocols (that have to carry an entire route). To limit energy con-

sumption, we add a *recent path* measure to avoid repeating the routing process. This saves energy and time not only for the source that created the path but also to any other sender that might be using a subset of the path that was previously established towards the destination. Switching to the recent path mode happens when two paths to the same destination intersect i.e. a packet passes through a node or a neighbor of node that has an established path to destination. 3DGR will terminate after traversing  $O(|n|)$  hops in worst case where  $n$  is the set of all nodes in the network.

5. **Battery Awareness:** Unlike what we used to believe, the energy consumed from a battery is not equivalent to the energy dissipated in the device. Based on a discrete time battery model, we present an optimization to 3DGR protocol to dynamically schedule routing in sensor networks. Our algorithm is aware of the battery status of network nodes and schedules recovery to extend their lifetime.
6. **Loop Free and Robustness:** 3DGR is inherently *loop-free*, since each data message propagates away from its source in a specific direction (as discussed in Thm. 4.1); it is *robust*, meaning that the data message can reach its intended destination by following possibly independent routes considering that every time we select the best path using an optimization function; 3DGR also tolerates inaccuracies in location information as illustrated in the simulation results. It provides a higher successful delivery ratio with high tolerance to localization errors.

The simplicity of this algorithm and the elimination of assumptions made previously by other routing algorithms make it suitable for real applications in sensor networks. The rest of the paper is organized as follows. Related research work is analyzed and summarized in Section 2. In Section 3, the basics of 3DGR are presented and the theoretical analysis of the routing algorithm is provided in Section 4. In Section 5, examples of different cases are provided and 3DGR is compared with GPSR analytically. Simulation results are presented in Section 6. We conclude this paper in Section 7.

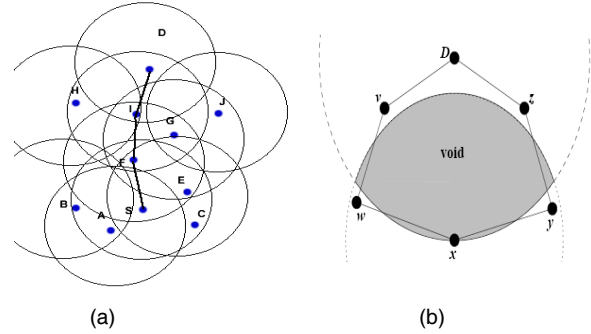
## 2 RELATED WORK

In WSNs routing, approaches that depend on either proactive routing, like dynamic Destination Sequenced Distance Vector DSDV[1], Optimized Link State Routing OLSR[2] or reactive routing, like Ad-hoc On demand Distance Vector AODV [3] still have significant problems with resources scarcity and communication overhead when the topology changes frequently due to mobility of nodes. Although approaches that are based on flooding or directional flooding like DREAM [4] have high robustness, they also have significant overhead resulting from flooding and may still fail when there are no nodes in the area in the direction of flooding within flooding angle. Ideas based on random walking like Rumor Routing [5] are limited in use to specific situations where events and

queries occurrences are within a specific range.

Another approach, geographical routing [6], has been proposed to be used as an alternative to flat routing algorithms in WSNs. Messages are not sent to designated devices, but rather to geographic locations. Some of these location based algorithms use restricted directional flooding like DREAM [4], Hierarchical approach like Terminodes [10] and Grid routing, Quroum systems like HomeZone and GLS [11], greedy approaches like Most Forward within R (MFR), Nearest with Forward Progress (NFP), and Compass Routing [12]. The greedy approaches provide efficient communication complexity of  $O(\sqrt{n})$  where  $n$  is number of nodes in the network (Fig. 1(a)). The main problem that faces greedy approaches is the void problem (Fig. 1(b)). The void problem arises when there isn't any node closer to the destination than the sender and thus results in the failure of the greedy approach in finding a path to the destination (although one might exist). Some face routing algorithms like GPSR [13] solve the void problem by the using the right hand rule; however, GPSR shares with all location based algorithms proposed so far the assumption that all nodes are roughly in a plane (i.e. the use of planer graphs). Such an assumption is not valid in real applications where nodes are distributed in three dimensional spaces [14]. Moreover, GPSR needs to build a planar graph using an algorithm like Relative Neighborhood Graph (RNG) or Gabriel Graph (GG) before the routing algorithm can be applied which results in extra overhead and less network lifetime. Kim et al [15] proposed another approach to remove non-planarities using cross link detection protocol CLDP.

Another face routing algorithm GOAFR [16] uses an ellipse to limit its searching radius for the next node on the path and keeps track of how far the packet has gone along the face and if no progress toward the destination is encountered, the packet is backtracked. Although GOAFR uses a different approach than GPSR, it still assumes that nodes must lie in a 2D plane. Also, it has been shown in [17] that the performance of GPSR decreases significantly with the increase in localization errors. Another drawback of GPSR is that packets follow boundary edges while traversing holes in the network which causes nodes on the boundary to be depleted quickly. Recently, Funke and Milosavljevic propose MGGR algorithm [18] which is macroscopic variant of geographic greedy routing. MGGR performs better than GPSR with imprecise node locations. However, MGGR introduces the use of land marks in addition to the need to form planar sub-graphs. MGGR also has a higher average communication cost per message than GPSR.



**Fig. 1.** (a) A greedy forwarding scenario. Each node selects the closest neighbor to destination. (b) The void problem in greedy forwarding.  $x$  is closer to  $D$  than its neighbors  $w$  and  $y$ . Although two paths,  $x-y-z-D$  and  $x-w-v-D$  exist to  $D$ ,  $x$  will not choose to forward to  $w$  or  $y$  using greedy forwarding.

An approach to solve void problem without planarization has been suggested by Liang et al [19]. The proposed algorithm, GDSTR, handles void problem by switching to route on a spanning tree that is likely to make progress toward the destination until it reaches a node where greedy routing can be continued. Although the building of a planar graph is avoided, GDSTR needs to build a spanning tree and each node needs to maintain information about the area covered by the tree below each of its tree neighbors and thus resulting in extra overhead.

To overcome some of the problems that arise due to the use of actual coordinates like localization errors, the use of virtual coordinates has been proposed [20-22]. In virtual coordinates, nodes' locations are specified relative to some reference fixed nodes. This reduces problems resulting from localization errors but requires the flooding of initialization packets from the reference nodes in order for other nodes to compute their relative positions. On the other hand, this makes the system vulnerable to signal fading during the initialization phase. Also, the conventional void problem is replaced by another void problem of the same nature when the node is closer to the destination than all its neighbors even using relative coordinate's measures. Moreover, some nodes may have identical virtual coordinates although they may be far apart.

#### Related Work in 3D Routing

More recently Durocher *et al* [23] show that routing in three dimensions is harder than routing in two dimensions and that it is possible to lift a two dimensional plane only to a limited extent. Also, they show that there aren't any previously proposed algorithms that guarantee packet delivery in three dimensional spaces. In 3DGR, the algorithm proposed in this paper, geographical routing is applied to three dimensional spaces and we show that if two nodes in the network are connected then 3DGR will be able to guarantee the delivery of packets between them (Thm. 4.3). Another very recent work includes Flury *et al* [29] where the authors consider the problem of 3D geographic routing in wireless ad hoc networks. They were interested in local, memoryless routing algorithms where each node bases its routing decision solely on its local view of the network. They show that a cubic routing

stretch constitutes a lower bound for any local memory-less routing algorithm, and propose and analyze several randomized geographic routing algorithms which work well for 3D network topologies. Earlier work in 3D routing includes 3D position based routing by kao *et al* in [28]. In this paper, a heuristic using the projective approach for face routing in 3-D was proposed. Their approach does not guarantee packet delivery as a planar graph cannot be extracted from the projected graph using only its local information before projection.

Unlike other approaches, we do not assume radio ranges are uniform and that they cover unit balls. Hence, we overcome problems and restrictions that are evident in previous geographical algorithms (there is no need to build a planar graph as in GPSR or MGGR). The major routing algorithms for sensor networks and their drawbacks are listed in **Table I**. In the next section, we provide the basics behind our routing algorithm.

### 3 ALGORITHM DESCRIPTION AND ANALYSIS

In this section, we start by explaining the techniques that will aid us in devising an energy efficient 3D routing al-

gorithm. These include: a *3D geocasting* technique, the addition of a *recent path* measure, *blacklisting*, and the concept of *battery awareness*. This is followed by a general description of 3DGR for the initialization phase as well as the sending and receiving phases. A detailed flowchart and pseudo-code of the routing are also presented. Finally, the complexity analysis of 3DGR is discussed.

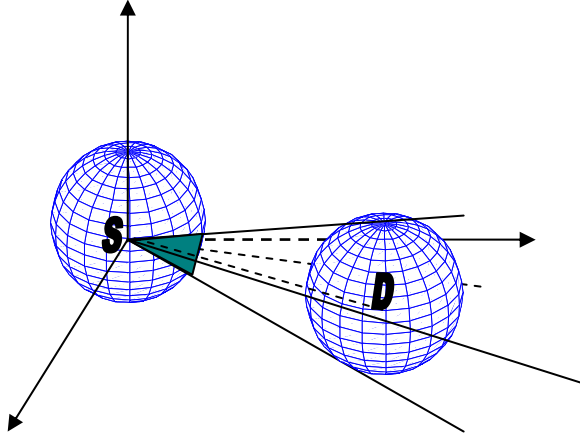
#### 3.1 A 3D Geocasting Technique

The purpose of geocasting is to send a message to nodes in a specific geographical region. Unlike directional flooding techniques, like DREAM [4], which flood packets in some direction, we do not use flooding in our algorithm. We use geocasting to limit the local broadcast of the small request packets to a region within angle  $\alpha$  in the direction of the destination. Hence, nodes that will respond to the request are those that are in the range of the sender and within an angle  $\alpha$  in the direction of the destination. Then the algorithm will select the best neighbor, using an optimization function discussed in Section 3.3, and forward the packets to that neighbor only.

The 3D geocasting problem is reduced to checking whether a point belongs to specific region in a three dimensional space.

**TABLE I.**  
DISADVANTAGES OF MAJOR ROUTING ALGORITHMS

Category	Abbreviation	Major Disadvantages
Proactive	<i>DSDV</i> <i>OLSR</i>	a) Maintenance of unused path occupies significant memory. b) Extra overhead if the topology changes frequently.
Reactive	<i>AODV</i>	a) It performs route discovery before sending packets which will result in extra delays for the first packets to be transmitted. b) Significant amount of overhead when the topology changes frequently. c) Packets on the route are likely to be lost if the route to the destination changes.
Greedy	<i>MFR</i> <i>NFP</i> <i>Compass</i>	a) May fail to find a path even though one might exist ( <i>the void problem</i> ). b) The position of the destination should be known with accuracy of one hop transmission range.
Restricted Directional Flooding	<i>DREAM</i>	a) Requires that all nodes maintain the position information of every other node in the network. b) The Communication complexity is $O(n)$ where $n$ is the number of nodes in the network. c) Least scalable and thus it is inappropriate for large scale networks. d) The redundancy of the packets received will result in wastage of power.
Hierarchical	<i>Terminodes</i>	a) Complex to implement. b) Requires the sender to know about specific positions leading to destination. c) Sender includes a list of positions in the packet header ( <i>extra overhead</i> ). d) Needs to check at regular intervals whether the path of positions is still valid or can be improved.
Hierarchical	<i>Grid</i>	a) Complex to implement. b) May fail in cases where the Terminodes succeeds in finding a path.
Geographical	<i>GPSR</i> <i>GOAFR</i>	a) Considers topologies where nodes are roughly in a plane. b) Needs to form planar graphs and thus resulting in extra overhead.
Geographical	<i>GDSTR</i>	a) Consider topologies where nodes are roughly in plane. b) Needs to form a spanning tree. c) Needs to maintain information about area covered below each of its tree neighbors.



**Fig. 2.** The geocasting region is the intersection of the 3D ball of center S and radius  $R_c$  ( $R_c$  is the communication radius of S) and the cone whose head is the sender and head angle  $\alpha$  (the shaded area).

To simplify the visualization of the problem, we consider that the range of the node to be a 3D ball – although this technique works with any arbitrary shape of radio range – by taking advantage of the fact that projection preserves order i.e. if a point  $(x, y, z)$  is in the region bounded by any three dimensional shape then its projection on any plane belongs to the projection of that shape on that plane. Without any loss of generality, our 3D region is the intersection of the ball representing the range of the sender – the source and every intermediate node is considered as a sender – and the cone whose head is the sender node and head angle  $\alpha$  is specified to suit the application as depicted in Fig. 2. When a node receives a request, we get the equation of the line (SD) between the source and the destination. A node  $P(x_p, y_p, z_p)$  belongs to the set of nodes within the geocasting area if it satisfies two conditions:

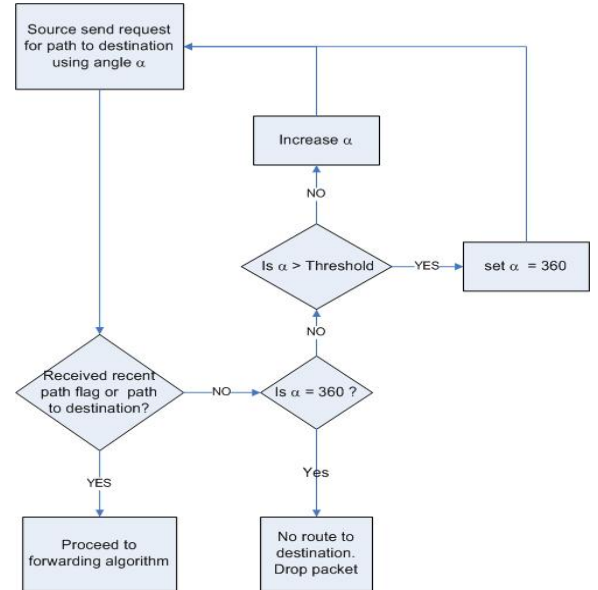
- *Condition (1):*  $P$  is on the same direction of the destination  $D$  according to the sender  $S$ . This can be verified by checking that  $P$  and  $D$  are on the same side of the line perpendicular to  $(SD)$  and passing through  $S$ .
- *Condition (2):*  $P$  is on the same direction of the destination  $D$  according to the sender  $S$  with respect to the line passing through  $S$  and making an angle  $\alpha$  with line  $(SD)$ . This can be verified by checking the following condition:  $d(P, P') \leq d(S, P') \cdot \tan(\alpha)$ .

where  $S$  and  $D$  are the locations of the sender and destination nodes respectively and  $P'$  is the projection of  $P$  on  $(SD)$ . If there is no node in the targeted region, the source node will not receive a reply and it will therefore increase the head angle  $\alpha$  and will resend the request packet. This will increase the targeted region to include more nodes. If no node replies, the angle  $\alpha$  is increased until it reaches a

threshold which would result in the request being locally broadcasted. A flowchart explaining the 3D geocasting techniques is provided in Fig. 3.

### 3.1.1 Choice of the threshold and $\alpha$ -increments

The threshold is used to skip the normal increments of  $\alpha$  and change to a local broadcast of the request. This is done when incrementing  $\alpha$  and resending the request will cost more than locally flooding the request. The choice of  $\alpha$  and the method used for increments depend mainly on the density of the network. If the density is high then  $\alpha$  should be chosen to be small to conserve energy (less nodes will respond however the number of nodes is enough). If the density is low then  $\alpha$  should be chosen to be large to get more nodes to respond. The same strategy is applied for increments. If the density is high, a small increment in the angle will include a significant number of new nodes whereas when the density is low a large increment is needed to include enough new nodes.



**Fig. 3.** Flowchart of the Geocasting Algorithm

### 3.2 A Recent Path Measure

To decrease the routing overhead, recent paths to destinations are maintained locally and temporally. Initially nodes have no recent paths for any destination. When a node wants to forward a packet, it uses the routing algorithm described in sections 3.4-3.6 to select the next node to which the packet will be forwarded. When the packet is forwarded, the sender node adds a recent path flag to its list of recent paths specifying the destination and the next node on the path. Thereafter, whenever the sender wants to forward a packet to the same destination, the packet is forwarded directly to the next node on the path without the need of applying the routing algorithm to select the next node and hence saving a significant amount of energy and minimizing the overall end-end delay. Considering that each node has limited storage, the storage of recent

paths information is done using a dynamic link list. Recent paths are added dynamically for each destination whenever a node forwards a packet towards a specific destination and the path is removed for a recent path list when it expires after a specific time. This makes the size of the list very efficient where it will be limited to the number of destinations during a recent path expiry interval. Hence, even if source-destination pairs are chosen randomly in the network (for example in a node to node communication case), the size of the recent paths list is limited to the number of destinations during one expiry interval. This eliminates the problem of using a large recent path buffer. In extreme cases, when there are too many destinations during one recent path interval, a recent path interval is simply decreased and recent paths are updated more frequently.

The expiry interval of the *recent path* can be updated dynamically by the routing algorithm depending on several factors. Monitoring the list of neighbors is one of the factors that can be used in updating the expiry interval. When the list of neighbors is updated frequently as in highly mobile networks, the interval is reduced to accommodate for the dynamic nature of the network. Another factor is the frequency of change in the list of destinations. When the rate of changing destinations is very high to the extent that the recent path buffer cannot accommodate such a change, the interval is decreased to reduce the buffer size since the benefits of using a recent path measure will not be evident. The size and the rate of data transfer also play an important role in estimating the expiry interval taking into consideration the energy and battery state of the neighboring nodes. When the data packets are large and the rate is high, nodes on the routing path get depleted faster and the interval is reduced to result in a better load distribution over the whole network. An example of the recent path measure is shown in Figure 4.

### 3.3 Blacklisting

In order to prevent looping in the routing algorithm, a blacklisting technique is used. A blacklist record has a structure similar to that of a recent path. When a node wants to blacklist a neighbor as the next node for a specific destination, it adds a record indicating that. When a node receives some possible paths to a destination from its neighbors, it excludes neighbors who have been blacklisted.

### 3.4 Battery Awareness Optimization

#### 3.4.1 Background

Recent study in battery technology helps us better understand the battery behavior [24]. Unlike what we used to believe, the energy consumed from a battery is not equivalent to the energy dissipated in the device. When discharging (Fig. 5(b)), batteries tend to consume more power than needed, and can reimburse the over-consumed power later. The process of the reimbursement is often referred to as battery recovery (Fig. 5(c)). This behavior is due to chemical characteristics of batteries. The battery consists of two electrodes, anode and cathode, separated by electrolyte. When the battery is connected to a load,

electrons start to flow from the anode to the cathode and an oxidation-reduction reaction occurs. With continuous discharge of battery, the outer surface of the cathode becomes reduced and the outer surface of the anode becomes depleted of electrons. Hence, the oxidation reduction reaction is stopped. This causes the battery to disconnect from the load although it still has some energy in it (Fig. 5(e)). On the other hand, if the battery is given the chance to perform redistribution of electrons between the outer and inner surfaces (Fig. 5(c)), an operation referred to as *recovery*, the battery will be able to use the energy still stored in it (Fig. 5(f)).

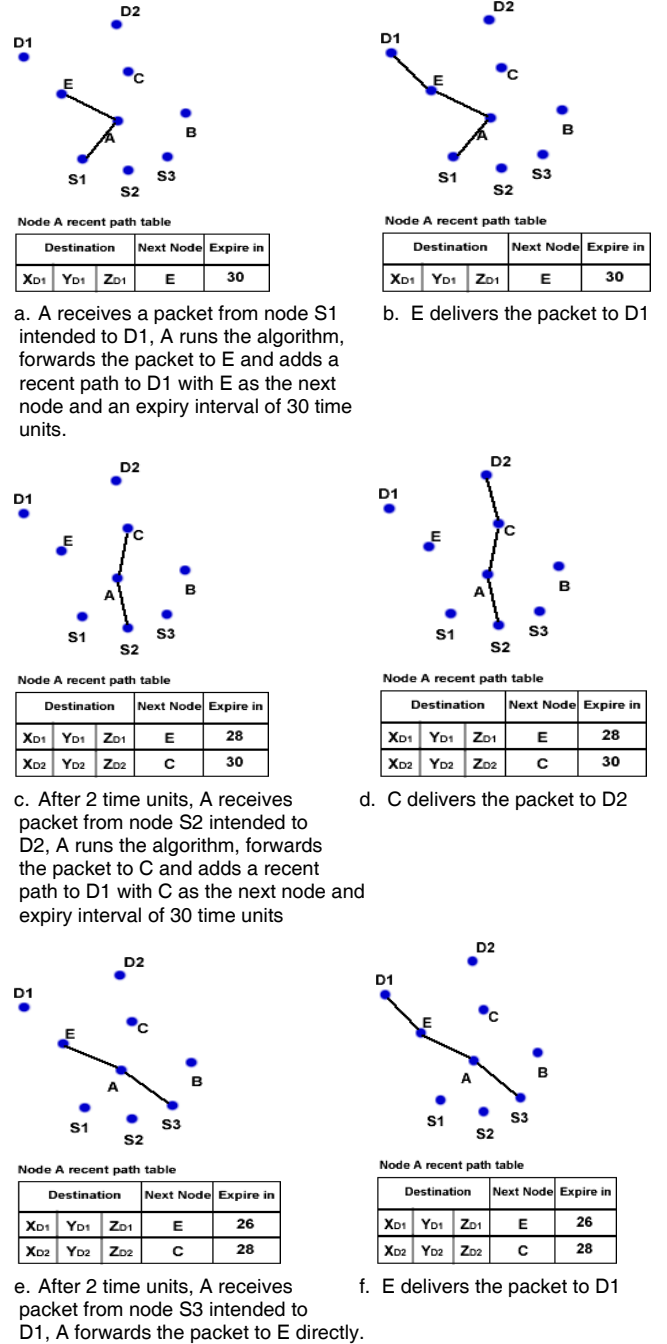


Fig. 4 Recent path at node A with different sources and destinations

### 3.4.2 Incorporation of the Battery Awareness in the Routing Algorithm

Based on a discrete time battery model, we present an optimization to 3DGR protocol to dynamically schedule the routing in sensor networks. Our distributed routing algorithm is aware of the battery status of the nodes and schedules recovery to extend their lifetime. We evaluate the performance of our routing algorithm with and without the battery awareness optimization in the simulation results.

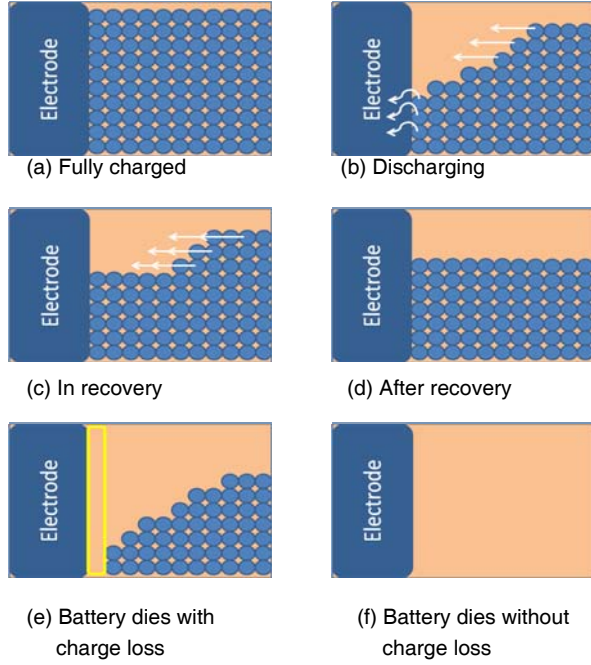


Fig. 5. Battery at different states.

The nature of network traffic as packets allows us to assume a discrete model for the battery life time. Several battery models have been discussed in literature. A good discussion of battery models can be found in [25]. In this paper, we use the term *battery state* to refer to the recovery state of the battery and the term *battery energy* to refer to the energy still stored in the battery. Also, we create a simple model considering that the battery initially is fully recovered; the *battery state* will decrease with each packet being sent or received and would recover when the node is idle. We also consider that the rate of discharging and recovering is equal. The battery state after a time interval  $t_0$  is given by:

$$B_{t+t_0} = B_t \pm \left(\frac{E}{E_0} \times t_0\right)\% \quad (1)$$

where  $B_t$  is battery recovery state at time  $t$ ,  $E$  is battery energy at the current time,  $E_0$  is the initial battery energy, and  $t_0$  is the time interval. To incorporate battery awareness in our routing algorithm, we would like to select the sensor node  $S_i$  from the set of nodes that have replied to the request such that the following function is minimized:

$$f(S_i) = w_1 D_i + w_2 \frac{1}{B_i} + w_3 \frac{1}{E_i} \quad (2)$$

where  $w_1$  is weight assigned to distance factor ( $D$ ),  $w_2$  is weight assigned to the current battery state ( $B$ ) calculated from equation (1) and  $w_3$  is the weight assigned to the energy that is still stored in the battery ( $E$ ). The goal will be to select the next node that will minimize  $D$  while maximizing  $B$  and  $E$ .

### 3.5 Initialization Phase

When the nodes are initially deployed, each node will broadcast one HELLO packet which includes their position information and will schedule another HELLO packet to be sent at a random time. This random scheduling of the second HELLO packet is to reduce collisions of HELLO packets during the initialization interval where all nodes will be sending HELLO packets. When a node receives a HELLO packet, it checks if the sender is already in its list of neighbors. If it is not, it adds the sender to its neighbor list. It then checks if it is within the random time scheduled for the second HELLO packet (which means the node is still in the initialization phase) then it does not reply with a HELLO packet. If the node is not in the time scheduled for the second HELLO packet, then the HELLO packet received is from a new node added to the network; hence the node broadcasts a HELLO packet to inform the new node about itself. If the sender is already in the neighbors' list, it silently drops the packet. To overcome HELLO packets getting lost, request packets in the sending and receiving phases are used as additional mechanism to add nodes to the neighbors' list. This use of the HELLO packets allows the adaptation to the addition of nodes to the network easily. We will also see the mechanisms used to adapt to node failures later.

### 3.6 Sending and Receiving Phase

When a source wants to send a packet to some destination, it starts by checking if it has a recent path to that destination. If such a path exists, the packet is forwarded to the next node in the path. Otherwise, it geocasts (using some angle  $\alpha$  used to suit the application) a small request packet that includes the coordinates of the destination. Also, the sender will set a timer  $R_t$ .

When a node receives a request packet, it checks if the sender is already in its neighbors' list. If not, it assumes that it has missed the HELLO packet sent by this neighbor during the initialization phase and therefore adds it to its neighbors' list. Each node that has heard the request checks if it is in the intended region specified by the request packet. If not, it silently drops the packet. Otherwise, it checks for a recent path to the requested destination (the time interval in which a path is considered recent is specified to suit the application and environmental conditions). If a recent path exists, it sends a

response to the request indicating the recent path. Otherwise, it checks its neighbors' list and selects the best candidate using the evaluation function from equation (2). Then, it sends a response for the request specifying the closest distance to the destination it can reach, the estimated cost of energy to reach there and the status of its battery (only nodes that have heard the request packet will reply hence node failure will be detected automatically).

Type	This id	This			Destination		
		X	Y	Z	X	Y	Z
tan alpha	project of line between this and destination on XY plane			on XZ plane			
	slope	constant		slope	constant		

(a) The format of a REQUEST packet.

Type	This id	This		
		X	Y	Z

(b) The format of a HELLO packet.

Type	This id	Recent flag	Measure

(c) The format of a REPLY packet.

Type	Originating node			Destination		
	X	Y	Z	X	Y	Z
Sender ID	Reciever ID		TTL	Backtrack		
DATA						

(d) The format of a DATA packet.

**Fig. 6.** Different formats of the packets used in the routing algorithm. Some fields that are used during the routing process include: Type: packet type (HELLO, REQUEST, REPLY, or DATA), This: node which is sending the packet, Alpha: The geocasting angle, measure: the value of function (2).

When the timer  $R_t$  expires, the node checks if the replies it had received contain a recent path leading to the packet being forwarded on that direction. If there is no recent path then it selects the best path using the evaluation function and forwards the packet only to the next node on the best path chosen. This eliminates the possibility of having duplicate packets. If no neighbor replies to the request packet (either there is no neighbors in that direction or those neighbors suffer a void problem), then the geocasting angle is increased and the process is repeated again. This gives one more chance to nodes that were included in the previous casting in case there were some difficulties during the last transmission.

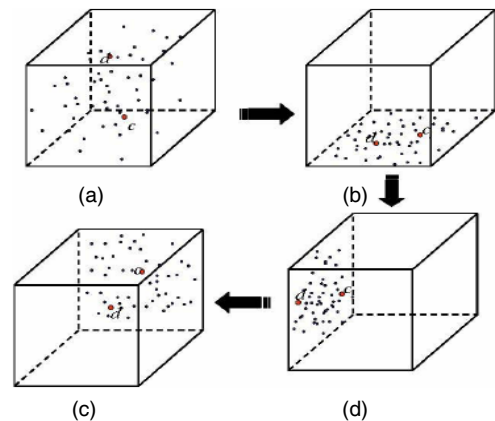
When the geocasting angle reaches the threshold, the node locally broadcasts the request and hence all neighboring nodes will respond. The sender then selects the

best path and forwards the packet to the next node on it (this forces the algorithm to try all possible paths in this case starting from the best one available). Preventing routing loops and pingponging is discussed in Section 4. When a node receives a packet, it checks if it is the destination node. If it is, then no forwarding is needed. Otherwise, it repeats the sending process described above. The flowchart and pseudo code of the routing algorithm is provided in Fig. 8.

Another approach that can easily be incorporated to our routing algorithm when the rate of topology changes in the network is very high, is to make a node that has a recent path to the destination send a request only to the neighbor on this path to make sure that it is still alive and in range. If a reply is received, then it forwards the packet. If no reply is received, then it goes back to the method of geocasting requests. This would ensure successful delivery even if the mobility is very high since for each packet, the node makes sure that next node is ready to receive the packet. This approach is of particular importance when obstacles and changes in the environment like weather conditions are present. In such cases, two nodes may be connected at a point in time and disconnected at another point of time. By sending the small request packets, the node verifies that the next node on the path can hear it before forwarding the packet. If the request is sent at a time when there is no connectivity, the node will select a new reliable path. This approach can be used when delivery of each packet is of very high importance and the mobility in the network is very high.

### 3.7 The algorithm in 2D

When applications that are not designed to understand the third dimension use 3DGR as a routing algorithm, 3DGR will automatically adapt and it will route using the projection of the nodes on the 2D plane. This is done easily since 2D is a special case of 3D where one of the coordinates is constant. For example, when the Z coordinates are constant; the nodes will be in  $xy$  plane as depicted in Figure 7 (b). The algorithm will route based on this fact and will perform its measures on the projection of the nodes on  $xy$  plane. The same argument holds for nodes in the  $yz$  or  $xz$  planes. Simulations of the routing algorithm in 2D are included in Section 5.



**Fig. 7** (a) 3D distribution of nodes (b) Projection in the  $xy$  plane



*Sending Pseudo-code*

```

source = originating node

while source != destination
     $\alpha$  = initial value
    Path = NULL

    while  $\alpha \leq$  threshold
        geocastRequest( $\alpha$ );
        battery = battery - energy/init_energy
        setTimer(Rt);

        while(! Rt expires )
            rp = receiveReply();
            battery = battery - energy/init_energy
            // if the source of the reply
            // is not in neighbors' list
            if(!isNeighbor(rp.source))
                neighbors.add(rp.source)
            end if

            if (rp.path is recent &&
                rp.energy > energy threshold)
                path = rp.path
            else
                // choose better path
                if(rp.path < path)
                    path = rp.path
                end if
            end if

        end while

        if(path != NULL)
            break;
        else if  $\alpha = 2\pi$ 
            // we have tried all directions
            break;
        else if  $\alpha >$  Threshold
            // try broadcasting
             $\alpha = 2\pi$ 
        else
            // increase geocasting angle
            if ( $2 * \alpha \leq 2\pi$ )
                increase  $\alpha$ 
            Else
                 $\alpha = 2\pi$ 
            end if
        end if

    end while

    if(path = NULL)
        // there is no path to destination
        dropPacket()
        break
    else
        // check if the packet is sent backward
        if(distance(path.next) > distance(this))
            pkt.backward = true
            pkt.backwardNode = this
        end if
        forwardPacket(path)
        battery = battery - energy/init_energy
    end if

end while

```

*Receiving Request Pseudo-code*

```

rqst = receiveRequest()
battery = battery - energy/init_energy
if (!isNeighbor(rqst.source))
    neighbors.add(rqst.source)
end if

if (inRegion())
    temp = rqst.destination
    // get best path from neighbors
    path = pickBestPath(neighbors)
end if

return path

```

*Receiving Data Pseudo-code*

```

data = receiveData()
battery = battery - energy/init_energy
if (destination is neighbor)
    forwardPacket(destination)
    battery = battery - energy/init_energy
if (local recent exists)
    if(pkt.backward && pkt.backwardNode = recent)
        blacklist(recent)
        call send(destination)
    else
        forwardPacket(recent.next)
        battery = battery - energy/init_energy
    end if
else
    call send(destination)
end if

```

*Battery awareness Pseudo-code (pickBestPath)*

```

pickBestPath(neighbors) {
    neighb = neighbors.first
    path = neighb.path
    while(neighbors.hasNext)
        neighb = neighbors.next
        D = neighb.distance
        // compare paths and pick the better one
        if(D < path.D)
            path = neighb.path
        end if
    end while
    return(w1*path.D + w2*this.battery + w3*this.energy)
}

```

*Geocasting Pseudo-code*

```

geocastRequest( $\alpha$ )
{ // get the equation of line from source to d
  // destination
  y = ax + b
  z = a'x + b'
  // send parameters with geocasting angle
  send (a, b, a', b',  $\alpha$ )
}

inRegion(){
  // get the projection point of the node on the
  // line y = ax+b and calculate z using z = a'x+b'
  (x', y', z') = proj(this, a, b, a', b')
  // check if node is in region
  if(distance(node, projection) < distance(source,
    projection) * tan  $\alpha$ ) // node is in region

```

Fig. 8. (b) The pseudo-code of the routing algorithm.

## 4 THEORETICAL ANALYSIS

The following definitions aid the mathematical formulation of the three dimensional routing problem.

### 4.1 Definitions

- $P(S_x, D)$ : the set of all possible paths from node  $S_x$  to destination  $D$
- $p$ : a path to destination  $D$  that belongs to  $P(S_x, D)$
- $n$ : the set of all nodes in the network
- $N_x$ : the set of neighboring nodes for a node  $S_x$
- $R_{C_x}$ : the set of recent paths stored at node  $S_x$
- $r_c(S_{xi}, x, y, z)$ : a recent path record  $\in R_{C_x}$  to destination  $(x, y, z)$  where the next hop is  $S_{xi}$
- $B_x$ : the set of blacklist records at node  $S_x$
- $b(S_{xi}, x, y, z)$ : a blacklist record blacklisting node  $S_{xi}$  as the next hop to destination  $(x, y, z)$
- $|A|$ : cardinality of set  $A$  i.e. number of elements in the set  $A$
- $d(a, b)$ : The Euclidean distance between points  $a$  and  $b$

### 4.2 Preventing looping and pingponging

**Theorem 4.1** *If there is a loop on a path from the source to the destination  $D(x, y, z)$ , then 3DGR will always detect and avoid that loop automatically.*  $\square$

**Proof** To prove this theorem, we take advantage of the fact that for a loop to occur, the packet must be forwarded in the direction opposite to the greedy choice at some point i.e. away from the destination. Suppose that the packet forwarding was in the opposite direction at node  $S_{xi} \in N_x$ . When  $S_x$  forwards the packet to  $S_{xi}$ , it adds a recent path  $r_c(S_{xi}, x, y, z)$  to its  $R_{C_x}$ . Since  $S_{xi}$  will forward the packet to a node  $S_{im} \in N_{xi}$  where  $d(S_{im}, D) > d(S_{xi}, D)$ , it adds a flag indicating itself as the node where backward forwarding started. If the packet ever reached  $S_x$  again,  $S_x$  will find that  $r_c(S_{xi}, x, y, z) \in R_{C_x}$  and that  $S_{xi}$  has indicated that it sent the packet backwards. Thus  $S_x$  will add  $b(S_{xi}, x, y, z)$  to  $B_x$  and send a request for a path. This means that  $S_x$  will pick the next hop from the set  $N_x - B_x$  and guarantees that the packet will be forwarded to a different path every time it comes back to  $S_x$ . This operation is done only during path discovery. Afterwards, the recent path will indicate the right direction directly.  $\square$

### 4.3 Correctness and Complexity

**Lemma 4.2**  $\forall S_x \in n$ , if  $P(S_x, D) \neq \emptyset$  then 3DGR will be able to find the next node  $S_y$ :  $S_y \in p$  where  $p \in P(S_x, D)$ .

**Proof** In the worst case,  $(|P(S_x, D)| = 1, |N_x| > 1) \wedge \exists! S_{x+1} \in N_x : S_{x+1} \in p$  i.e. there is exactly one path from node  $S_x$  to  $D$  and only one node in the neighborhood of  $S_x$  belongs to this path. Then we have two cases:

- First, the best local choice  $S_i$  of the algorithm – based on the criteria specified by formula (3) – is  $S_{x+1} \in p$ . In this case  $S_{x+1}$  will be picked up in the first round and the condition is satisfied.
- Second, the best local choice  $S_i \neq S_{x+1}$ . In this case the packet will be forwarded to  $S_i$  until the algorithm anticipates a void problem. Then the algorithm tries to find an alternative path and since  $|P(S_x, D)| = 1$  the

algorithm has no option other than backtracking to  $S_x$ . Loops will be avoided as explained in theorem 4.1. This process is repeated for each  $S_i \in N_x$  until it finds  $S_i = S_{x+1}$ .  $\square$

**Theorem 4.3** If  $\exists p \in P(S_x, D)$  then 3DGR will successfully deliver packets from  $S_x$  to  $D$ .  $\square$

**Proof** To prove this theorem we use loop invariant approach.

- Initializing phase: the packet is at  $S_0$  then by lemma 4.2 the packet will be forwarded to  $S_1 \in p$ .
- Intermediate nodes: Our loop invariant is that 3DGR will forward the packet to next node  $\in p$ . Initialization phase guarantees that  $S_1 \in p$ . By applying lemma 4.2 on  $S_1$ , 3DGR will forward the packet to  $S_2 \in p$ . Since any path  $p \in P(S_x, D)$  has a finite number of nodes in it, repeated application of lemma 4.1 guarantees delivery of the packet to destination  $D$ .  $\square$

**Lemma 4.4** If  $\exists p \in P(S_x, D)$ , 3DGR will deliver packets to the destination after traversing  $O(|n|)$  hops in the worst case otherwise disconnection is reported.  $\square$

**Proof** If  $\exists p \in P(S_x, D) \rightarrow$  Lemma 4.2 guarantees that  $\forall S_x \in n$ , 3DGR will be able to find  $S_{x+1}$ :  $S_{x+1} \in p$ . The worst case is  $|P(S_x, D)| = 1, |N_x| > 1, \exists! S_{x+1} \in N_x : S_{x+1} \in p$  and  $S_{x+1}$  is the worst local choice for node  $S_x$ . This means that node  $S_x$  will check all  $S_i \in N_x$  before it tries  $S_{x+1}$ . The use of blacklisting exclude the possibility of trying the same path twice since a node that has blacklisted will not be used again and nodes that have blacklisted their neighbors will not respond to requests for a path to the destination. This means that at most all nodes of the network will be traversed with additional cost of backtracking steps i.e. the asymptotic cost is  $O(|n|)$ . If  $|P(S_x, D)| = 0$ , the packet will reach a node where all neighbor nodes have been tried and blacklisted and disconnection is detected.  $\square$

**Lemma 4.5** If  $\exists p \in P(S_x, D)$  and nodes in the network are uniformly distributed then 3DGR will deliver packets to the destination by traversing  $O(\sqrt[3]{|n|})$  hops in the average case where  $n$  is set of all nodes in the network.  $\square$

**Proof** If nodes are uniformly distributed then the network will have a ball shape with radius  $\sqrt[3]{|n|}$ . In the average case the distance between any two random nodes in the network is the radius of the network. As proven in theorem 4.3, 3DGR will be able to find  $p \in P(S_x, D)$ .

- If greedy forwarding works then the average cost will be  $\sqrt[3]{|n|}$ .
- If a void problem is encountered and since no path will be tried twice as explained in lemma 4.4, the cost will increase by a constant multiple and becomes  $c\sqrt[3]{|n|}$  where  $c$  is the number of wrong paths tra-

versed, however, the asymptotic cost remains  $O(\sqrt[3]{n})$ .  $\square$

**Lemma 4.6** If  $P(S,D) \neq \emptyset$ , then, on average, the number of hops traversed by 3DGR to reach the destination is  $O(p^*)$  where  $p^*$  is the number of hops in the shortest path.  $\square$

**Proof.** Number of hops on the path picked by 3DGR,  $p$ , depends of the topology of the network as number of hops in the optimal path,  $p^*$ . We consider the case of uniformly distributed nodes and the same proof can be generalized to any case. Such networks have ball shape and the average distance between any two nodes is the radius of the network  $\sqrt[3]{n}$ . Lemma 4.5 proves that in uniformly distributed network 3DGR delivers the packet with cost of  $O(\sqrt[3]{n})$ . The constant multiple,  $c$ , that exists between  $p$  and  $p^*$  is at maximum while establishing the path for the first time since wrong paths maybe tried; however,  $c$  is reduced for subsequent packets.  $\square$

## 5 EXAMPLES AND COMPARISON

In this section, we provide in depth comparisons and examples illustrating some of the fundamental features of 3DGR.

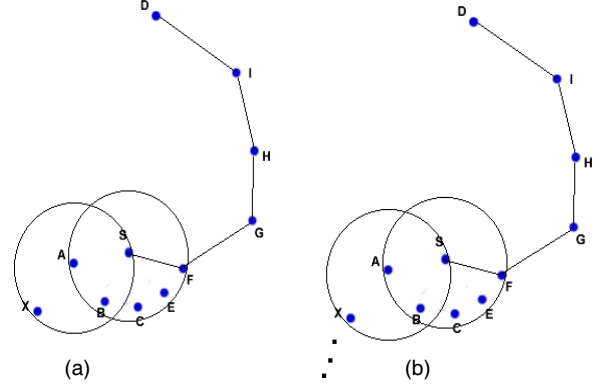
### 3DGR vs. GPSR

When packet forwarding follows a normal greedy approach and the void problem is not encountered, all greedy algorithms will have the same performance in terms of the number of hops traversed. On the other hand, algorithms differ in their approach to overcome the void problem. GPSR, for example, solves the problem by forming a planar graph and following a right hand rule to forward the packet. This method succeeds in finding a path to destination when one exists. However, it may incur significant overhead and in some cases has longer delay, or even fail if the TTL (*Time To Live*) expired before the packet is forwarded to the correct neighbor. An example showing this case is given in (Fig. 9(a)).

When GPSR faces the void problem, it switches to perimeter mode and starts to forward the packet using the right hand rule. This means that the packet will be forwarded to node A which will face a void problem and will forward it to B and the same problem is faced by B which will forward it to C. In C, there is a neighbor E which is closer to the destination than C itself so it will follow the greedy approach. E will forward it to F and then F to G-H-I-D. It is obvious that even if routing loops were avoided, a lot of unnecessary forwarding is done since the packet could have been sent from the source S to F from the beginning. The problem of GPSR becomes even worse if A had a neighbor that is farther than A and precedes B while using the right hand rule. In that case, the packet will be forwarded in the wrong direction and it will take a longer time to come back to the right path and there is also a possibility of the packets being dropped because the TTL might have already expired (Fig. 9(b)).

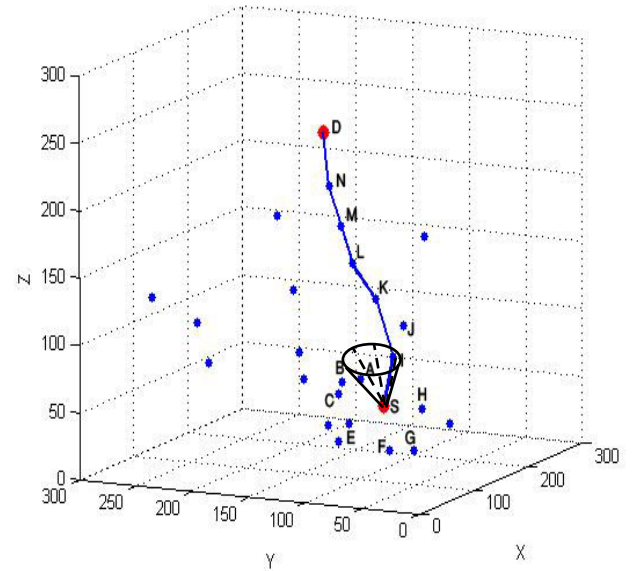
On the other hand, 3DGR will send a small request

packet before it forwards the initial packet. When it receives no answer, it increases the geocasting angle. Node F will eventually reply with a possible path to the destination and node A will reply with a path with a higher cost. Hence, the packet is forwarded directly to the node F and the path traversed is S-F-G-H-I-D.



**Fig. 9.** (a) GPSR will do extra forwarding using perimeter mode. (b) Situation becomes worse for GPSR with the presence of  $x$

A more detailed example of 3DGR in three dimensions is presented in Fig. 10. Suppose that node S wants to send a packet to node D. S checks if it has a recent path to destination D. Since this is the first packet, no such path exists. Hence, it sends a small request packet with angle  $\alpha$  (let us say  $\alpha = 30$  degrees). It sets its timer  $t$  and waits for responses from neighboring nodes that are located within a cone whose head angle is 30 and base is the circle with its center lying on the line (SD).



**Fig. 10** Example showing the routing algorithm in 3D. The path taken is S-A-I-K-L-M-N-D.

## 6 SIMULATION RESULTS AND ANALYSIS

For the purpose of simulation, Network Simulator NS-2 is used. Simulations are divided into two parts. First, we simulate our algorithm in three dimensions to see its behavior in a three dimensional environment. Secondly, we simulate our algorithm in two dimensions to see how the algorithm behaves when used with applications that do not understand the third dimension. Also, we use simulations in two dimensions to compare with other famous geographical routing algorithms such as GPSR and GOAFR.

### 6.1 3D Simulations

#### 6.1.1 3D Simulation Environment

Simulation is done on a network of 100 randomly deployed nodes in  $100 \times 100 \times 100$  cube. The communication range of each sensor node is 40 m. We adopt byte division for sending and receiving energy (i.e. energy in sending is calculated as energy for sending/receiving one byte times number of bytes to be sent/received). Idle listening also consumes some energy that is significantly lower than sending and receiving energy. We also consider that the node has 50 joules initially. Five source-destinations pairs are chosen randomly and the results are based on the average of 100 simulation runs of the algorithm. Each source generates a packet of 500 bytes every two seconds. Metrics used in the evaluation are: tolerance to localization errors, energy consumption, network lifetime with and without energy/battery awareness, and the overall end-to-end delay. Due to the lack of existing work in 3D geographical routing, we chose to compare our results to Kao et al's work [28] (PFR) where they proved that their projective 3D face routing algorithm gives significantly better delivery rate than the other proposed greedy routing algorithms in 3D.

#### 6.1.2 3D Simulation Results

a. **Localization errors:** Geographic routing in wireless sensor networks is based on the prerequisite that every node has information about its current position. In our simulation, we induce errors on locations of the nodes randomly using the node's communication range as a measure for the position deviation. Hence, a 10% deviation means that there is a localization error in the node position which is 0.1 the node's communication range. We assume that all the nodes do not know the exact positions of any other node in the network including the sink. As illustrated in Fig. 11, unlike PFR, 3DGR has a delivery ratio close to one when the localization errors are below 25%. Packet delivery ratio decreases as the location deviation increases; however, 3DGR will still have a good delivery ratio (around 80%) even when position deviation is 100%. The high tolerance to lo-

calization errors is mainly because 3DGR does not use the exact locations to route the packets. Instead, it uses the location information to forward packets in the right direction.

- b. **Energy consumption:** Energy is taken as the average energy per node calculated over intervals of 25 seconds. As shown in Fig. 12, unlike PFR, 3DGR has efficient energy consumption. The slope becomes steeper as the nodes start to originate packets (around 25 sec). This is because there are no existing paths to the destinations and therefore, new paths are being established. Then, the slope decreases since recent paths now are being used to forward the packets to the destinations.
- c. **Network lifetime:** The metrics used in evaluating system lifetime is the number of active nodes after a period of time. The overall lifetime is the continuous operational time of the system before the percentage of active nodes drops below a specified threshold (for example 90%). For evaluating the battery awareness in our algorithm, we use our battery optimization function in choosing the best path and we assign equal weights for the distance and battery factors and more weight on the energy factor. As can be seen in Fig. 13, the incorporation of energy/battery awareness in the path selection criteria increases the lifetime of the network significantly with a gain of 80%. This is mainly due to better distribution of the load over the whole network. This also shows that 3DGR can effectively incorporate energy and battery awareness while maintaining a high performance. Note that when 25% of nodes die, the network loses its connectivity and the packets cannot be routed to destinations.
- d. **End-to-End Delay:** As shown in Fig. 14, 3DGR has a small end-end delay on average. The initialization phase takes longer time since routes are established for the first time; however, 3DGR delay becomes much smaller afterwards. When the recent record updating parameter is activated, the delay graph shows a pulse every time the recent record is updated. The overall effect is that the average end-to-end delay slightly increases. On the other hand, this increases the adaptability to node mobility and links failures. This parameter can be set to suite the desired application based. If the expiry interval is set to be small, then the delay will increase but the adaptability to changing topology increases as well – suitable for networks where the topology changes frequently. If the expiry interval is set to be large, then the delay is decreased but the adaptability to the changing topology is also decreased – suitable for networks where topology rarely changes. So, the choice of the expiry interval should take into consideration the frequency of topology changes.

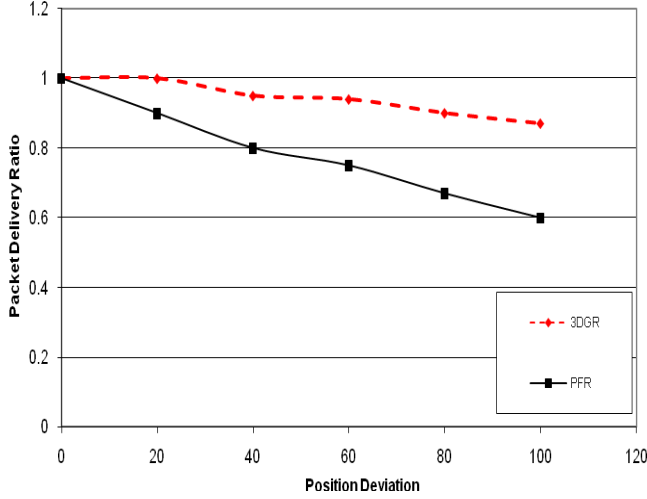


Fig. 11 Packet delivery ratio as a function of the location error

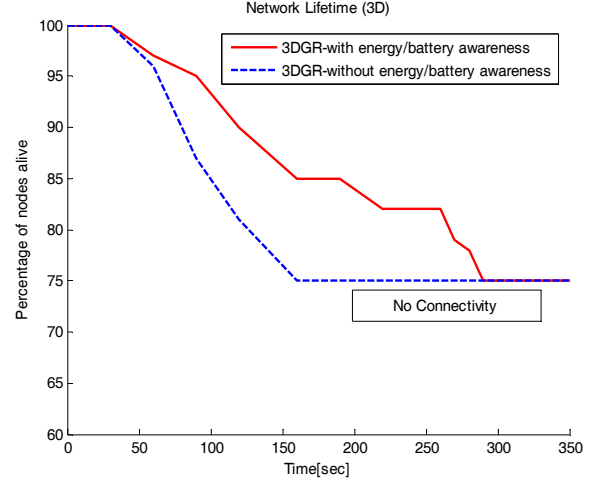


Fig. 13 The system lifetime using battery awareness feature

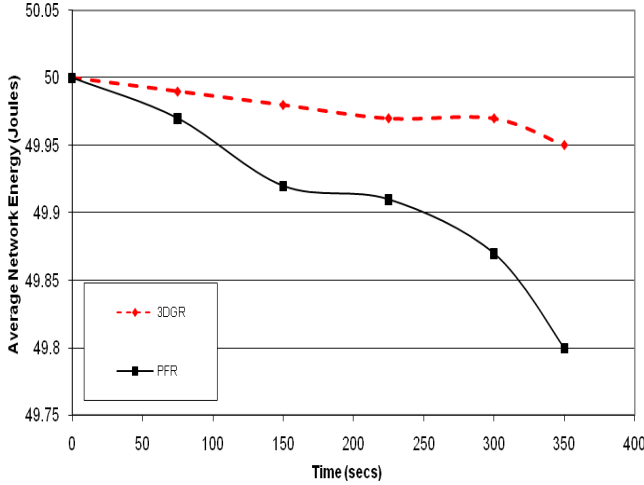


Fig. 12 Average energy per node as function of time

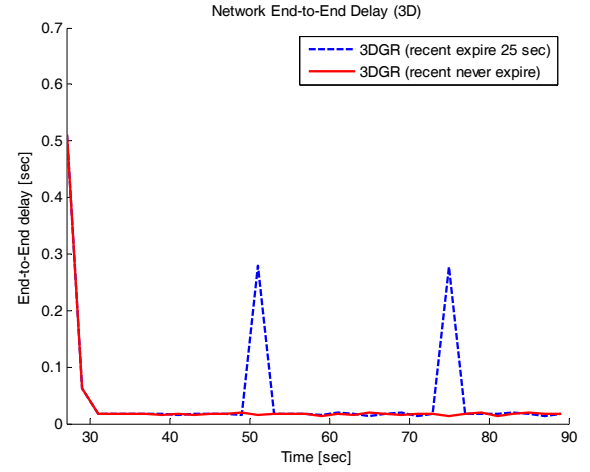


Fig. 14 End-to-End delay using 3DGR with and without activation of recent path.

## 6.2 2D Simulations

### 6.2.1 Simulation Environment

Simulation is done on a network of 100 randomly deployed nodes in  $100 \times 100$  area. The communication range of each sensor node is 40 m. We also adopt byte division for sending and receiving energy. We also consider that each node has 50 joules initially. Five source-destinations pairs are chosen randomly and results are based on the average of 100 simulation runs of the algorithm. Each source generates a packet of 500 bytes every two seconds. Metrics used in the 2D evaluation include: energy consumption, end-to-end delay, tolerance to localization errors, and path stretch. Comparisons are also done with other well known geographical routing algorithms such as GPSR and GOAFR. We use path stretch as a uniform metric where we compare with most of the 2D geographical routing algorithms.

### 6.2.2 2D Simulation Results

- a. **Energy consumption:** As shown in Fig. 15, simulation results show that 3DGR conserves significant energy compared to GPSR. This is mainly because 3DGR uses small requests packets locally to pick the best path. Also when facing void problems, 3DGR picks the best path available by checking both directions whereas GPSR follows always the right hand rule although the best path may be in the other direction. This causes GPSR to be wrong (on average) half of the time. Moreover, GPSR continues to route each packet independently and hence loses more energy every time. On the other hand, 3DGR uses information from previous packets by the use of the *recent path measure* to avoid spending energy in re-discovering paths especially when the void problem is encountered.

- b. **End-to-end delay:** As shown in Fig. 16, the end-to-end delay for 3DGR is larger than that for GPSR during the initialization phase; however, the delay in 3DGR becomes much smaller afterwards. When the recent record updating parameter is activated, delay graph shows a pulse every time the recent record is updated, but the overall effect on average end-to-end delay is small and will remain lower than that of GPSR. As mentioned earlier, this increases the adaptability for node movements and links failures.
- c. **Localization errors:** The assumption of exact location information is inappropriate in real deployments since location information is gained either through GPS signals or some localization algorithm, both of which are error-prone. An evaluation of greedy forwarding algorithms and GPSR in the case of location errors can be found in [17, 26 and 27]. In plain greedy mode, a high packet drop rate due to false dead ends was observed. The drop rate increases with higher network density. In this experiment, we compare the effect of localization error of 3DGR to that of GPSR. We use percent of range as a unit for errors and we evaluate the packet delivery ratio on 0, 10, 25, 50, 75, and 100 % position deviations. As seen in Fig. 17, 3DGR has better delivery ratio than that of GPSR when the deviation in position is more than 10%. This difference increases as the percentage in the position deviation increases.
- d. **Path Stretch:** Path stretch is the ratio of the total path length to the optimal path length between any two nodes. We compare the path stretch of 3DGR to GPSR [13] using Gabriel Graph (GG) and Cross Link Detection Protocol (CLDP), GOAFR using CLDP [15], and GDSTR [19].

As shown in Fig 18, 3DGR has the best performance on various densities which means that 3DGR picks shorter paths to destinations than other algorithms do. This is because 3DGR picks the best path available locally based on two hops information (although information of only one hop neighbor is stored). Also, 3DGR routes packets based on the local direction of destination according to the sender. This leads to an automatic correction of the path in case the packet was forwarded in a non optimal direction. The difference in results of the various algorithms is mostly pronounced at critical densities when the void problems arise and each algorithm has to use its special technique to route the packet around the void region.

On the other hand, the performance of all the routing algorithms converges when using simple greedy forwarding would result in a close to optimal path. It is expected that after the peak, where the difference is mostly evident, the hop stretch will de-

crease with increasing density. However, in our simulation we see another smaller hump. This is because in our simulations a void case arose in one of the scenarios generated using this density. The hump is smaller because the number of void cases is smaller. Also, this case shows that even with higher densities when a void case might arise, 3DGR will outperform other routing algorithms. GPSR has the worst performance because it uses the right hand rule to route around the void region. This means that GPSR will forward packets in the wrong direction around 50% of the time (on average). GOAFR has a better performance because it uses an ellipse to limit the searching radius and increases the radius of the ellipse in case the first search failed. GDSTR has better performance than both GPSR and GOAFR due to the use of a tree to forward the packets. When the two forwarding directions are available, GDSTR picks the tree with the shortest path. However 3DGR outperforms both GDSTR and GOAFR because they handle the void problem after it occurs so they need to do extra forwarding whereas 3DGR handles the void problem before it occurs in most cases and thus we are able to avoid it.

## 7 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a novel 3D geographical routing algorithm that takes into consideration the special characteristics of wireless sensor networks and eliminate the assumptions made by earlier geographical routing algorithms. We show that 3DGR (with the ability of operating in 3D spaces) has better results when compared to other algorithms such as GPSR and GOAFR. Although 3DGR uses geographical information to route the packets in the direction of the destination, has a relatively high tolerance for localization errors and chooses a close to optimal path (if one exists). The incorporation of battery model leads to the extension of network life time and better distribution of loads.

Part of our future work would be to develop a 3D routing algorithm that can work underwater taking all the underwater challenges into consideration such as: high propagation delay, impaired channel due to fading, limited bandwidth, high bit error rate and failures because of fouling and corrosion. Experimenting with other battery models and optimizing the battery function will form another side of our future work.

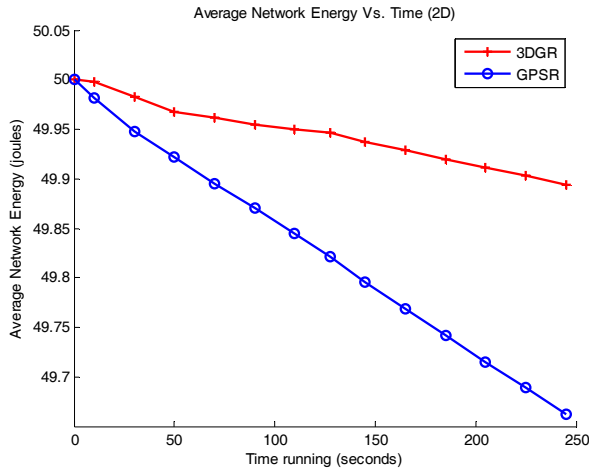


Fig. 15 Average energy per node as a function of time using 3DGR and GPSR.

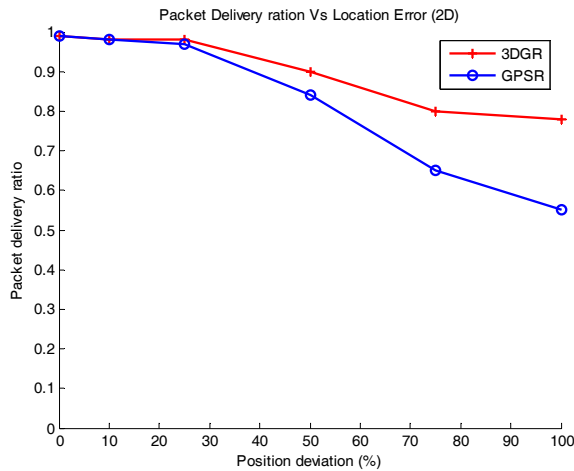


Fig. 17 Packet delivery ratio as a function of the location error

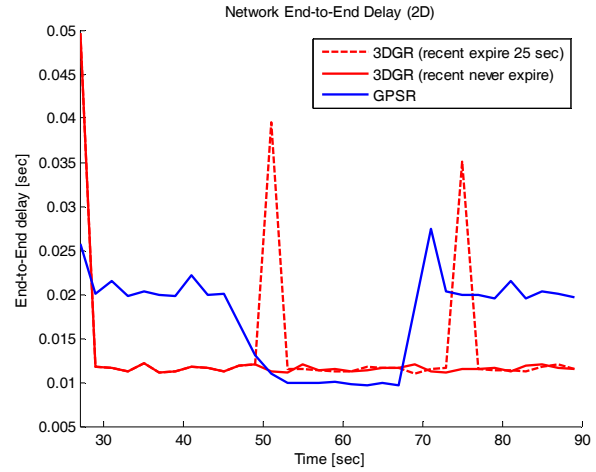


Fig. 16 End-to-End delay using 3DGR and GPSR. 3DGR takes longer time in initialization phase then it outperforms GPSR.

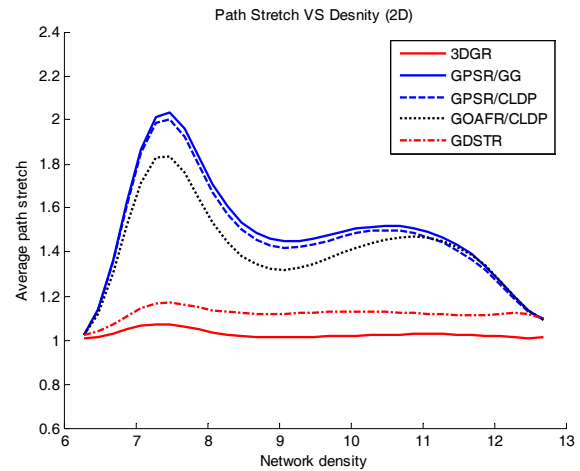


Fig. 18 Path stretch relative to network density. 3DGR outperforms other algorithm in picking optimal paths.

## REFERENCES

- [1] C. Perkins, and P. Bhagwat, "Highly-Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *SIGCOMM '94 Conference on Communications, Architectures, Protocols, and Applications* London, UK, pp 234-244, Sept. 1994.
- [2] Z. Ren, Y. Zhou, and W. Guo, "An Adaptive Multi-Channel OLSR Routing Protocol Based on Topology Maintenance," *IEEE International Conference on Mechatronics & Automation*, Niagara Falls, Canada, pp 2222-2227, Jul. 2005.
- [3] C. Perkins, and E. Royer, "Ad-Hoc On-Demand Distance Vector Routing," *2nd IEEE Workshop on Mobile Computing Systems and Applications*, LA, USA, pp 90-100, Feb. 1999.
- [4] S. Basagni, I. Chlamtac, V. Syrotiuk, and B. Woodward, "A Distance Routing Effect Algorithm for Mobility (DREAM)," *4th ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, TX, USA, pp 76-84, Oct. 1998.
- [5] D. Braginsky, and D. Estrin, "Rumor Routing Algorithm for Sensor Networks," *First Workshop on Sensor Networks and Applications (WSNA)*, GA, USA, pp 22-31, Sept. 2002.
- [6] M. Mauve, J. Widmer, and H. Hartenstein, "A Survey on Position-Based Routing in Mobile Ad-Hoc Networks," *IEEE Network*, pp 30-39, Nov./Dec. 2001.
- [7] NASA/JPL Sensor Webs Project, <http://sensorwebs.jpl.nasa.gov/>.
- [8] X. Hong, M. Gerla, R. Bagrodia, P. Estabrook, T. Kwon, and G. Pei, "The Mars Sensor Network: Efficient, Power Aware Communications," *IEEE Military Communications Conferences (MILCOM 2001)*, McLean, VA, Oct. 2001.
- [9] F. Akyildiz, D. Pompili, and T. Melodia, "Underwater Acoustic Sensor Networks: Research Challenges," *Ad Hoc Networks (Elsevier)*, vol.3, pp. 257-279, May 2005.
- [10] L. Blazevic, L. Buttyan, S. Capkun, S. Giordano, J.-P. Hubaux, and J.-Y. Le Boudec, "Self-Organization in Mobile Ad-Hoc Networks: the Approach of Terminodes," *IEEE Communications Magazine*, pp 39-45, June 2001.
- [11] J. Li, J. Jannotti, D. Decouto, D. Karger, and R. Morris, "A Scalable Location Service for Geographic Ad Hoc Routing," *6th Annual ACM/IEEE International Conference Mobile Computing and Networking (MobiCom)*, MA, USA, pp 120-30, Aug. 2000.
- [12] E. Kranakis, H. Singh, and J. Urrutia, "Compass Routing on Geometric Networks," *11th Canadian Conference on Computational Geometry*, Vancouver, BC, pp 51-54, Aug. 1999.

- [13] B. Karp and H. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *6th International Conference on Mobile Computing and Networking (ACM Mobicom)*, Boston, MA, USA, pp 243-254, Aug. 2000.
- [14] S. Commuri and M. Watfa, "Coverage Strategies in 3D Wireless Sensor Networks," *International Journal of Distributed Sensor Networks*, pp 333-353, Oct. 2006.
- [15] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker, "Geographic Routing Made Practical," in *Proceedings of NSDI 2005*, pages 217- 230, May 2005.
- [16] F. Kuhn, R. Wattenhofer, and A. Zollinger "Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing," *4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Annapolis, MD, 2003.
- [17] M. Witt and V. Turau, "The Impact of Location Errors on Geographic Routing in Sensor Networks," *International Conference on Wireless and Mobile Communications*, pp 76-81, Jul. 2006.
- [18] S. Funke and N. Milosavljevic, "Guaranteed Delivery Geographic Routing Under Uncertain Node Locations," *IEEE INFOCOMM*, pp 1244-1252, Anchorage, AK, USA, May 2007.
- [19] B. Liang, B. Liskov, and R. Morris, "Geographic Routing without Planarization," *USENIX Symposium on Networked Systems Design and Implementation*, Apr. 2006.
- [20] A. Caruso, S. Chessa, S. De and A. Urpi, "GPS Free Coordinate Assignment and Routing in Wireless Sensor Networks," *IEEE INFOCOMM*, pp 150-160, Miami, FL, USA, Mar. 2005.
- [21] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker and I. Stoica, "Beacon Vector Routing: Scalable Point-to-Point Routing in Wireless Sensornets," *NSDI*, 2005.
- [22] T. Moscibroda, R. O'Dell, M. Wattenhofer, and R. Wattenhofer, "Virtual Coordinates for Ad hoc and Sensor Networks," *ACM Joint Workshop on Foundations of Mobile Computing (DIALM/POMC)*, Philadelphia, PA, USA, Oct. 2004.
- [23] S. Durocher, D. Kirkpatrick, and L. Narayanan, "On Routing with Guaranteed Delivery in Three-Dimensional Ad Hoc Wireless Networks," *ICDCN*, pp 546-557, Kolkata, India, Jan. 2008.
- [24] R. Rao, S. Vrudhula, and D. Rakhmatov, "Battery Modeling for Energy Aware System Design," *IEEE Computer Society*, pp. 77-87, Dec. 2003.
- [25] C. Ma and Y. Yang, "Battery Aware Routing for Streaming Data Transmissions in Wireless Sensor Networks," *Mobile Networks and Applications*, pp 757-767, May 2006.
- [26] T. He, C. Huang, B. Blum, J. Stankovic, and T. Abdelzaher, "Range-Free Localization Schemes for Large Scale Sensor Networks," *9th Annual International Conference on Mobile Computing and Networking*, San Diego, California, Sept. 2003, pp. 81-95.
- [27] Y. Kim, J.-J. Lee, and A. Helmy, "Modeling and Analyzing the Impact of Location Inconsistencies on Geographic Routing in Wireless Networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 8, no. 1, pp. 48-60, Jan. 2004.
- [28] Kao G, Fevens T, Opatrny J. Position-Based Routing on 3-D Geometric Graphs in Mobile Ad Hoc Networks. *Proc.17th Canadian Conference on Computational Geometry (CCCG'05)*, Windsor, Ontario, Canada, 2005; 88-91.
- [29] Roland Flury and Roger Wattenhofer. Randomized 3D Geographic Routing. *27th Annual IEEE Conference on Computer Communications (INFOCOM)*, Phoenix, USA, April 2008.