GPS Free Coordinate Assignment and Routing in Wireless Sensor Networks

Antonio Caruso, Stefano Chessa, Swades De, and Alessandro Urpi

Istituto di Scienza e Tecnologie dell'Informazione,
Area della Ricerca, CNR di Pisa-S.Cataldo, 56100 Pisa, Italy.

Abstract— In this paper we consider the problem of constructing a coordinate system in a sensor network where location information is not available. To this purpose we introduce the Virtual Coordinate assignment protocol (VCap) which defines a virtual coordinate system based on hop distances. As compared to other approaches, VCap is simple and have very little requirements in terms of communication and memory overheads. We compare by simulations the performances of greedy routing using our virtual coordinate system with the one using the physical coordinates. Results show that the virtual coordinate system can be used to efficently support geographic routing.

Index Terms— Wireless sensor network, virtual coordinates, hop based greedy routing, performance analysis, simulations

I. INTRODUCTION

A sensor network is a specialized ad hoc network composed by a large number of low power, low cost nodes (also called sensors) [1]. A sensor comprises one or more sensing units, a processor and a radio transceiver, and is powered by an embedded battery. Sensors collect information about the surrounding environment (sensor field) and they self-organize into a wireless ad hoc network in order to exchange sensed data and to connect with external sink nodes that issue queries to the network.

Typical applications of sensor networks are environment sampling, monitoring disaster areas, health monitoring, surveillance, security, inventory management, and they have also been envisioned as an architectural support for applications of pervasive computing [2].

The effective development of scalable sensor networks presents a number of research challenges ranging from routing protocols to algorithms for data collection, fusion and stream processing. Current sensor networks employ a query distribution and data collection based on a simple model known as data diffusion [3], [4]. This model assumes that the sink node has a permanent connection with the network and performs most of the data analysis, while the role of the network is limited to data acquisition and, in some cases, to simple data processing. This assumption is motivated by the fact that, with the current technologies [5], sensors are unable to perform complex data processing and storage. However, in the effort of improving the management of data streams produced by sensor networks, it has recently been proposed to integrate database and sensor network technologies [6], [7], and the concept of data-centric storage has been introduced in [8]. The integration of database technologies with sensor networks, and the use of data-centric paradigms, requires support of efficient and robust routing protocols, more general than those used to support data diffusion.

To this purpose, routing protocols based on geographic information of the sensors [9], [10], [11] have been proposed as a viable alternative to existing routing protocols for wireless ad hoc networks such as [12], [13], in order to reduce the burden of maintaining routing tables in the sensors and avoiding the cost of route discovery.

Although geographic routing may contribute to reduce the routing overhead on the sensors, it requires that the sensors be aware of their physical position. This information can be obtained by equipping all the sensors with devices such as GPS, however, cheapest (and approximate) solutions may be obtained by equipping with GPS only a limited subset of sensors and using this information to infer the position of the other nodes. For this reason, the problem of inferring nodes location in sensor networks in which just a few (or no) nodes know their geographic position is of great practical interest. Proposed solutions are based on different assumptions on node capabilities and knowledge, and each offers a different result accuracy.

Traditional approaches were aimed at deriving coordinates resembling as close as the real ones, when services like GPS are not present. Measures of goodness of a solution are computational and communication complexities, precision and accuracy. A complete survey

of methods and solution to approximate real position of nodes is presented in [14].

Commonly, a few nodes (often called anchors) are assumed to know their exact position, by means of special hardware or because they were positioned in well known points, and consequently programmed. The other nodes infer their positions, using techniques like time of arrival, time difference of arrival [15], [16], angle of arrival [17], [18], signal strength [19], [20], or hop distance [21]. A survey can be found in [22].

In [23], [26] the requirements of the coordinate assignment are somehow relaxed. In fact, authors study how to find virtual coordinates that preserve some interesting properties of the network.

In this paper, we reconsider the approach of [26] and [21]. In particular we focus on sensor networks composed by a very large number of sensors, deployed with relatively high density in the sensing field, and where the sensors do not have any information about the network boundary and position. In this scenario we introduce a new *Virtual Coordinate assignment protocol* (called VCap) to support geographic routing which defines coordinates exclusively based on hop distances. As compared to [21], we do not attempt to approximate physical coordinates based on hop distances. Rather, we construct a virtual topology which is possibly unrelated to the physical topology of the network.

The VCap protocol exploits four rounds (based on broadcasts) to identify three anchor nodes (namely X, Y and Z), and it assigns each node with a triplet of coordinates given by the hop distances from the anchors. The storage overhead for each sensor is limited to the storage of the triplet of coordinates it is assigned.

We show that, for a given density of the network, this coordinate system is consistent, i.e., nodes with the same coordinates lie within a limited number of hops from each other, and we show by simulation that this coordinate system can efficiently support geographic routing.

The rest of the paper is organized as follows. Section II reports related works and Section III describes the Virtual Coordinate assignment protocol. In Section IV the protocol overhead is discusses. Correctness analysis of the protocol is presented in Section V. Simulation results are reported in Section VI and Section VII draws the conclusions.

II. RELATED WORKS

In [23], authors define a virtual coordinate system where coordinates do not need to be a precise representation of real ones: the authors assume that the nodes have only proximity information, i.e., every node knows the order of proximity to every neighbor, and they define a coordinate system that respect such a proximity. If the nodes also have information about real distances from their neighbors, coordinates may be enhanced in order to reflect real ones. With this approach anchor nodes are not necessary, but, if present, they can be used to improve the solution accuracy.

The coordinate assignment proposed in [23] is based on a multidimensional scaling (MDS) [24], [25], which is a technique to map a set of points in a multidimensional space to a space with another number of dimensions, using a distances matrix computed in the starting space. The coordinate assignment in [23] is composed by three steps:

1) Compute shortest paths between all pairs of nodes. When no distance information is known, length 1 is assumed for every link.
2) Apply MDS to the distance matrix, in order to get a 2D or 3D map respecting distances.
3) Given sufficient anchor nodes, map relative coordinates into real ones (if needed).

The main drawbacks of this protocol are its complexity (in a network with n nodes the first step have complexity $O(n^3)$) and, most importantly, that it is based on a centralized approach which is not quite feasible in a wireless sensor network scenario.

To overcome these drawbacks, a distributed protocol of coordinate assignment with lower complexity has been introduced in [26]. This solution, which exploits graph embedding [27], aims at assigning the nodes with *virtual coordinates*, i.e., coordinates unrelated to real ones, which, however, can efficiently support geographic routing algorithm such as GPSR [9].

In [26] the authors introduce a family of coordinate assignment protocols with increasing complexities suitable for different scenarios.

In the first scenario, the nodes positioned on the borders of the network know their physical position. Every node iteratively computes its coordinates as the average of its neighbor's coordinates until it converges toward a virtual coordinate. Simulations show that after 1000 iterations, a geographic routing relying on these coordinates achieves a success ratio over the 99% and the average path length is only slightly worse than when real coordinates are used. A similar performance can be achieved if only some of the perimeter nodes are aware of their positions.

In the second scenario, the perimeter nodes know they lie on the network border, but they are not aware of their physical position. In this case the protocol runs a preliminary round in which the hop distances

among any pair of border nodes are discovered. To this purpose, each perimeter node floods the network with a beacon containing a hop counter. When a perimeter node receives all the beacons, it fills a *perimeter vector* with the distances between itself and all the other perimeter nodes. In a second stage each perimeter node sends its perimeter vector to all the other perimeter nodes, which can then fill a hop-distance matrix containing the hop distances between any pair of perimeter nodes. In turn, the distance matrix is used to compute the virtual coordinates of the perimeter nodes. Since non-perimeter nodes learn distances between perimeter nodes before starting the protocol, they can also compute approximate initial coordinate for themselves. Simulations show that this approach is rather effective.

The authors also consider the case in which the nodes do not have any information about the network boundary nor they have information about their physical position. This case is managed using two bootstrap nodes: the sensors execute a preliminary distributed protocol to evaluate the hop distance from the bootstrap nodes, then the sensors which are at a maximum distance from the bootstrap nodes in their neighborhood assume to be in the border of the network. Simulations show that, if the network is dense enough, just a few interior nodes take the wrong decision, and that the virtual coordinates are accurate enough to achieve a high success rate after a few iterations.

Although the approach of [26] leads to a coordinate system which can efficiently support geographic routing, the coordinate assignment protocol can be quite complex. In particular, in the scenario where the perimeter nodes have knowledge of their physical location, the proposed protocol requires a large amount of iterations to converge (in their example more than 1000 iterations). On the other hand, in the cases where the nodes are not aware of their physical location, the perimeter nodes should store the hop-distance matrix which have a size in the order of n (where n is the number of nodes in the network), and each perimeter node should perform $O(\sqrt{n})$ broadcasts to communicate to all the other perimeter nodes its perimeter vector.

III. The Virtual Coordinate Assignment protocol (VCap)

We consider a sensor network composed of a large number of nodes uniformly scattered in a sensing field. The nodes are assumed static, or they have very low mobility with respect to signal propagation speed. Every node has the same transmission range, and each node is able to communicate with all nodes within its range. For the sake of simplicity we also assume that every node has

a unique ID. This is used to break ties in the protocol, however this assumption can be easily removed as will be discussed later.

The coordinate assignment protocol has four phases, each of which selects one anchor node. The first anchor W is only used to enanche the chance of right selection of the other three anchors X, Y, and Z. At the end of the four phases each node is assigned with a triplet of coordinate (x, y, z), where each coordinate represents the hop distance of the node from one anchor.

The protocol proceeds as follows:

1) **Election of W:** Once the network is deployed, the sink node (or any other node programmed for this purpose) generates a W_SET message containing an hop counter initially set to 1. The counter will be called the w coordinate and its value is incremented by the forwarding nodes. If a node receives more than one message, it will consider (and forward) only the one with smaller w.

2) **Election of X:** This phase initiates after all nodes have been assigned with the w coordinate (To this purpose the nodes waits for a time larger than the time required to propagate the W_SET message in the network). The nodes that have the maximum value of w within a two-hop neighbourhood (in case of parity the nodes that have the maximum ID) decide they are in the boundary and generate an X_ELECT message containing their ID and an hop counter initially set to 1. This counter will be called the x coordinate and its value is incremented by forwarding nodes. If a node receives more than one message, it will consider (and forward) only the one with smaller w and, in case of equal w, the one with the larger[1] source ID. Note that also in this case the X_ELECT is generated after a random time in order to reduce the number of messages generated in the network.

3) **Election of Y:** This phase is similar to the Election of X. The difference here is in the rule used to select the nodes egligible to become Y, which selects the nodes that within a two-hop neighbourhood have maximum x, $w > \gamma$ for some parameter γ (which will be discussed in the rest of this section) and in case of parity have the maximum ID generate a Y_ELECT message which will set up a y coordinate. The same rules as before apply.

4) **Election of Z:** This phase is similar to the previous with the difference that only the nodes whose x, y coordinates satisfy some given rule $\phi(x, y)$ (which will be discussed below), and whose w
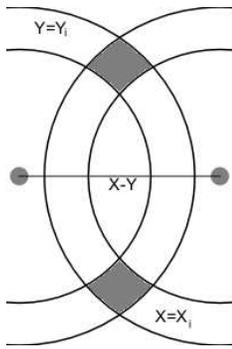
---

[1]Any other tie breaking rule may be used.

Fig. 1. If only two anchors are used (X,Y), nodes within the gray areas receive the same coordinate (X_i, Y_i).

Fig. 2. An example of possible anchor alignment. Without an appropriate rule to distance anchor Y from W, anchor Z may be aligned with X and Y.

is maximum within a two-hop neighborhood, will generate a Z_ELECT message, which will set up a z coordinate.

It should be observed that node W is used to let X, Y, and Z be close to the boundary of the network. This is achieved since only the nodes with maximal coordinate w and thus (with high probability) close to the network boundary are eligible as X, Y, or Z. After Z has been established, W is no more needed.

Note also that the ID of the sensors is used to break the ties in the protocol. However, if the sensors are anonymous, each sensor could generate an ID as a random number chosen on a range large enough in order to have a probability close to 0 that different sensors choose the same ID.

During the protocol execution, at high enough densities, coordinates propagate as circular coronas centered on the initiator node. In particular, the first hop resembles a circle with radius equal to the communication range (or better, the intersection between the circle and the network area). The ith hop resembles a circular corona with inner radius equal to $i-1$ times the communication range, and outer radius equal to the inner one plus the communication range (again, the intersection of such a corona with the network area).

When the last phase is completed, each node will use the assigned triplet (x, y, z) as coordinate. However, in general, the same coordinates will be shared by a set of nodes. Hereafter we will call a set of nodes with the same coordinate as a *zone*.

With this coordinate system, a geographic routing protocol delivers a packet to its destination zone. Then the packet is delivered to the desired destination node using a proactive ID-based approach.

For this reason, in order to efficiently support geographic routing the zones should span a limited number of hops.

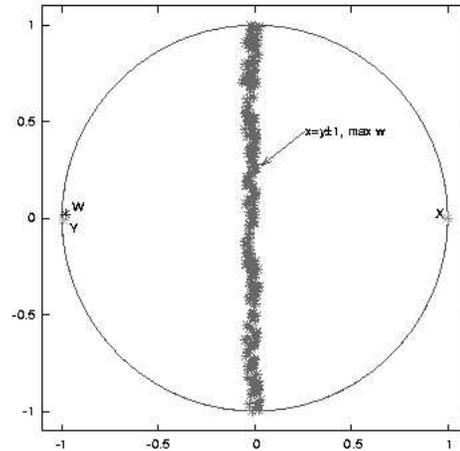This is also the reason for which all the three co-ordinates x, y, and z are necessary. In fact, if only two of them were used (say x and y), there would exist physically distant nodes, symmetric to the directrix connecting X and Y, which would share the same virtual coordinate. (see Figure 1 for an example). Thus, coordinate z is needed in order to break the symmetry.

As will be proved more formally in Section V, to enforce consistency of the coordinate system, the best choice for Z would be a node orthogonal to the center of the XY directrix. Thus, Z must be near to the network border, and almost equidistant from X and Y.

The rule $\phi(x, y)$ used in the last protocol phase to choose Z is a heuristic to find a node satisfying this last constraint. In our experiments, the rule selects nodes with maximum w and $x = y \pm 1$. Such a rule gave good results in simulations.

However, there is still a case in which ϕ is not enough to select only nodes on the network border. In fact, if W is on the network border, X will be chosen with high probability on the opposite border. If only distance from X was considered in order to select Y, it would be possible to have Y very near to W. In this case, the zone with nodes with $x = y \pm 1$ and maximum w would be quite large, and could include also nodes aligned with X, Y and W (Figure 2). For this reason the constraint $w > \gamma$, for some constant γ, used in the third protocol phase, ensures that Y is not aligned with X and W.

IV. VCAP OVERHEAD

In this section we discuss the communication and memory overheads of VCap. First we study the communication overhead considering the four phases of VCap. The first phase uses a broadcast originated by one node (typically the sink node) to identify nodes located on the

boundary of the network. In practice the effect of the first phase and the rule used to select nodes egligible as X is similar to the technique used in [26] (using two bootstrap nodes) to locate the perimeter nodes. Still referring to the protocol used in [26], in the case where nodes don't know their location, we can compare the stages 1,2 (a triangulation phase among perimeter nodes) with phases 2,3 and 4 of VCap. However, while in [26] each perimeter node should broadcast some information to all the other perimeter nodes, in VCap it is sufficient that only one perimeter node broadcasts some information to all the other nodes. This means that, in practice, VCap may require the same number of broadcasts as the first two steps of [26] in the worst case, but it may require a smaller number of broadcasts on the average.

To reduce the number of broadcasts required in phase 2 some well known technique on trading time for communication can be used. A simple protocol requiring one broadcast per phase is as follows. Let us assume that an upper bound of the time needed to propagate a message troughout the network t is known and that the nodes have integer IDs ranging from 0 to $n-1$ where n is the number of nodes in the network.

After having received the W_SET message from the sink node, node i waits for a time equal to $(n-i) \cdot 2t$ before generating its X_ELECT message. However, if it receives an X_ELECT message from another node with a value greater than or equal to w, it avoids to generate its X_ELECT message.

Let t_0 be the time at which the sink generates the W_SET message, and j be the largest ID of the nodes egligible as X. Then j generates its X_ELECT message at most at time $t_j = t_0 + (n-j) \cdot 2t + t$, where the last term accounts for the largest time that the W_SET message may take to reach j. Note also that the X_ELECT generated by j will reach any other node by the time $t_j + t$.

It is easy to see that any other node egligible as X and thus with an ID k with $k < j$ would generate the X_ELECT message at least at time $t_k = t_0 + (n-k) \cdot 2t \geq t_j + t$, and thus it will not generate the X_ELECT at all.

This means that this simple protocol would require only one broadcast per phase but in the worst case it may take a time close to $n \cdot 2t$ to complete.

In practice, an effective approach is to delay the generation of the X_ELECT messages for a random time chosen within a given range. Clearly, a large range has the effect of reducing the number of broadcasts but it would also increase the latency of the protocol.

Another issue is the memory overhead. The protocol presented in [26] requires each perimeter node to store a distance matrix, i.e., a matrix containing the distances between any arbitrary pair of perimeter nodes. A rough approximation of the number of perimeter nodes is $O(\sqrt{n})$, which implies that the distance matrix have a size $O(n)$. This is clearly an heavy burden if the protocol is applied in sensor networks. Furthermore, although the size of the distance matrix could be reduced by preventing some perimeter node to participate to the protocol, this solution does not appear to scale well with n. On the contrary, the memory overhead of VCap is constant as it requires each node to store only its assigned values of $w, x, y,$ and z.

V. PROTOCOL CORRECTNESS ANALYSIS

The VCap protocol presented in the previous section does not avoid that the two nodes have the same triplet of coordinates. Thus, instead of dividing the network in points, it causes a division in zones, and the nodes within a zone are labeled with the same virtual coordinates. The size of such a zone, depends on the node density and position of the anchors X, Y, and Z. As we have already noticed, in order to efficiently support geographic routing, size of the zones should be bounded. In this section we will show that the position of the anchors should be such that if two nodes are assigned the same virtual coordinates then they should belong to a bounded physical area, the size of which depends on the network density. First, we prove that such a bounded zone indeed exists under the assumption that the minimum hop distance between two nodes is somehow related to their Euclidean distance. Next, we will show that such a relationship existis.

A. Effect of anchor positions on zone of ambiguity

We define the network density as the average number of neighbors per node, and we make the following assumption:

Assumption 1: Consider two nodes at minimum hop distance h, there exist two values $l(h)$ and $u(h)$ such that the Euclidean distance d_E between the two nodes is bounded, i.e., $l(h) \leq d_E \leq u(h)$. The quality of the bounds depends on the network density Δ. In particular for each $h > 0$ holds:

$$\lim_{\Delta \to \infty} u(h) - l(h) = r$$

where r is the transmission range of the sensors.

If Assumption 1 holds and a node has the virtual coordinate (x, y, z), anchor X (respectively, Y, Z) must be positioned within a circular corona centered on the node itself, with inner radius $l(x)$ ($l(y)$, $l(z)$) and outer radius $u(x)$ ($u(y)$, $u(z)$) (Figure 3(a)).
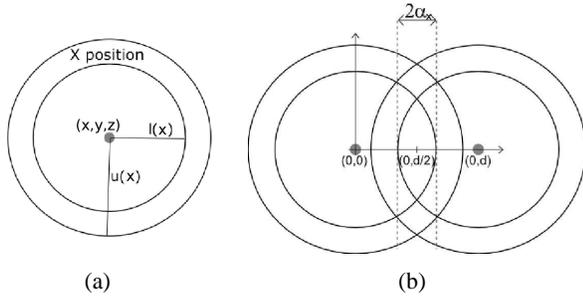
Fig. 3. Position feasibility zone for X and the intersection of two feasibility zones of two nodes sharing the same coordinate.
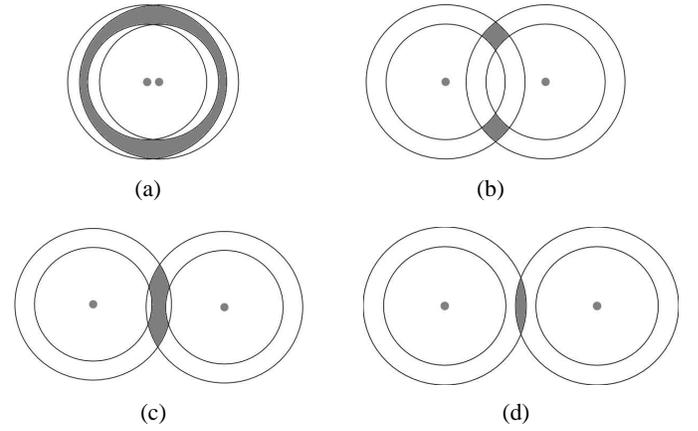

Fig. 4. Intersection between position feasibility zones of anchor X: Four possible cases depending on the distance between nodes sharing the same x coordinate.

Let us consider two nodes n_1 and n_2 which are assigned with the same virtual coordinate (x, y, z). We will show a bound to the physical distance between n_1 and n_2.

Without loss of generality, we assume that the physical coordinate of n_1 is $(0, 0)$ and that the physical coordinate of n_2 is $(d, 0)$, hence n_1 and n_2 are positioned at Euclidean distance d from each other (Fig. 3(b)).

Since the two nodes have the same x coordinate, the anchor X must be positioned in the intersection of the circular coronas previously described. Let us define d_1 (d_2) be the Euclidean distance between n_1 (n_2) and X. Then it follows that $l(x) \leq d_1 \leq u(x)$, and $l(x) \leq d_2 \leq u(x)$, and four cases may happen (the same cases also hold for Y and Z):

a) $0 < d \leq u(x) - l(x)$: when the nodes' distance is less than the uncertainty on X position (Figure 4(a)), then X itself lies on a feasibility zone whose size is bounded by the circular corona.

b) $u(x) - l(x) < d \leq 2l(x)$: in this case (Figure 4(b)), there are two feasibility zones, equidistant from n_1, and n_2.

c) $2l(x) < d \leq l(x) + u(x)$: in this case (Figure 4(c)), there is a unique feasibility zone for X.

d) $l(x) + u(x) < d \leq 2u(x)$: also in this case (Figure 4(d)), there is a unique feasibility zone for X.

Letting (x_x, y_x) be the physical coordinates of anchor X, it follows that, in all cases, it will be $d/2 - \alpha_x \leq X_x \leq d/2 + \alpha_x$ for some α_x (Figure 3(b)). Hereafter, we define the *feasibility zone* for X (respectively, Y, Z) as the area such that $x \in [d/2 - \alpha_x, d/2 + \alpha_x]$.

In case b) holds the following lemma.

Lemma 1: If $u(x) - l(x) < d \leq 2l(x)$ then $\alpha_x = (u(x)^2 - l(x)^2)/2d$.

Proof: As shown in the trapezoid drawn in Figure 5, the width of the feasibility zone (i.e., $2\alpha_x$) is the size of the upper base of an isosceles trapezoid having lower base size d, sides size $l(x)$, and diagonals size $u(x)$.

With some simple trigonometric considerations, it is possible to find the angles θ and ϕ, obtaining:

$$\theta = \arccos\left(\frac{l^2(x) + d^2 - u^2(x)}{2dl(x)}\right)$$
$$\phi = \pi - \theta$$

With the cosine rule for triangles, it is possible to obtain the upper base width by solving the following second degree equation:

$$(2\alpha_x)^2 + \frac{l^2(x) + d^2 - u^2(x)}{d}(2\alpha_x) + l^2(x) - u^2(x) = 0.$$

Defining $a = u^2(x) - l^2(x)$, the equation assumes the more compact aspect:

$$(2\alpha_x)^2 + \frac{d^2 - a}{d}(2\alpha_x) - a = 0$$

This second degree equation has two solutions, given by $2\alpha_x = a/d$ and $2\alpha_x = -d$. Since we are interested in the positive solution, the upper base width of the trapezoid is $\frac{a}{d} = \frac{u^2(x) - l^2(x)}{d}$. Hence we can conclude that $\alpha_x = (u(x)^2 - l(x)^2)/2d$. ∎

With an argument similar to that used in Lemma 1 it can be concluded that in case (c) holds $\alpha_x = (u(x)^2 - l(x)^2)/2d$, while in case (d) holds $\alpha_x = u(x) - d/2$.

Lemma 2: Let us suppose anchors X, Y, and Z form a triangle whose minimum height is H. Then if two points at Euclidean distance d share the same coordinates (x, y, z) (with, say, $x \geq y$ and $x \geq z$), it must hold either

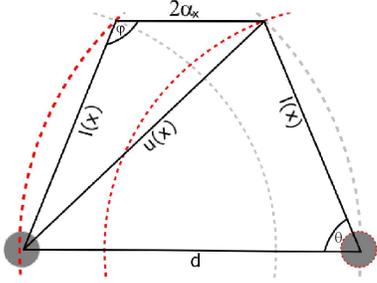$$d \leq \max\left(u(x) - l(x), \frac{u^2(x) - l^2(x)}{H}\right) \text{ or } H \leq u(x) - l(x).$$

Fig. 5. Determining the width of feasibility zone.

Proof: Without loss of generality we assume that:

$$u(x) - l(x) \geq u(y) - l(y) \quad \text{and} \quad (1)$$
$$u(x) - l(x) \geq u(z) - l(z)$$

and we consider the four cases shown in Figures 4(a)-4(d).

a) If $0 < d \leq u(x) - l(x)$ then the Lemma is trivially true.

b) If $u(x) - l(x) < d \leq 2l(x)$, let (x_x, y_x) be the physical coordinates of anchor X. Thus, from Lemma 1, the feasibility area for anchor X is delimited by: $d/2 - \alpha_x \leq x_x \leq d/2 + \alpha_x$, where $\alpha_x = (u(x)^2 - l(x)^2)/2d$.

   Note also that from 1 the feasibility zones for Y and Z are included in the feasibility zone for X. Then, since $max(x_x, x_Y, x_z) \leq d/2 + \alpha_x$ and $min(x_x, x_Y, x_z) \geq d/2 - \alpha_x$ it must hold (see Fig 3(b)): $|x_x - x_Y| \leq 2\alpha_x$, $|x_x - x_z| \leq 2\alpha_x$, and $|x_z - x_Y| \leq 2\alpha_x$.
   But, since the minimum height of the triangle whose vertices are X, Y, and Z is H, it must also hold: $max(|x_x - x_Y|, |x_x - x_z|, |x_Y - x_z|) \geq H$. From which $2\alpha_x \geq H$.
   Since $2\alpha_x = \frac{u^2(x) - l^2(x)}{d}$ follows that $d \leq \frac{u^2(x) - l^2(x)}{H}$.

c) If $2l(x) < d \leq u(x) + l(x)$, exactly the same argument as b) applies.

d) If $u(x) + l(x) < d \leq 2u(x)$, let (x_x, y_x) be the physical coordinates of anchor X. Thus, from Lemma 1, the feasibility area for anchor X is delimited by: $d/2 - \alpha_x \leq X_x \leq d/2 + \alpha_x$, where $\alpha_x = u(x) - d/2$.
   As in case (b), from $x = max(x, y, z)$ we have that the feasibility zone for Y and Z is included in the feasibility zone for X, and in particular it must hold: $|x_x - x_Y| \leq 2\alpha_x$, $|x_x - x_z| \leq 2\alpha_x$, and $|x_z - x_Y| \leq 2\alpha_x$ and, by the hypotheses of this Lemma: $max(|x_x - x_Y|, |x_x - x_z|, |x_z - x_Y|) \geq H$.

Since $2\alpha_x = 2u(x) - d \leq u(x) - l(x)$, it follows $H \leq u(x) - l(x)$.

■

Note that the bound of Lemma 2 does not depend on the network size. Furthermore, we deduct that the best position for the three anchors is the one that maximizes H. In particular, as will be shown in the following theorem, if the network is circular with diameter D, then the best placement of X, Y and Z is on the vertexes of an equilateral triangle inscribed in the circle.

Theorem 1: Let us consider a sensor network in a circular space of diameter D, and assume that the anchors X, Y and Z are placed on the vertices of an equilateral triangle inscribed in the circle of diameter D. If Assumption 1 holds then the maximum distance between two nodes in the same zone d satisfy:

$$\lim_{\Delta \to \infty} d = \frac{8}{3}r$$

Proof: Since the anchors X, Y and Z are placed on the vertexes of an equilateral triangle inscribed in the circle of diameter D, the minimum height of the triangle is $H = \frac{3}{4}D$.

From Lemma 2 it holds that either $d \leq max(u(x) - l(x), \frac{u^2(x) - l^2(x)}{H})$ or $H \leq u(x) - l(x)$.

Observe that, provided that $r < \frac{3}{4}D$, from Assumption 1, for a sufficiently large $\bar{\Delta}$ it holds: $u(x) - l(x) < \frac{3}{4}D = H$. Hence must be:

$$d \leq max\left(u(x) - l(x), \frac{u^2(x) - l^2(x)}{H}\right).$$

Notice that, if $u(x) - l(x) > \frac{u^2(x) - l^2(x)}{H}$, then from Assumption 1 the Theorem is trivially proved.

Then we consider the case:

$$d \leq \frac{u^2(x) - l^2(x)}{H} = \frac{(u(x) + l(x))(u(x) - l(x))}{H}$$

Since $u(x) \leq D$ (i.e., the distance of X from any node is trivially bounded by the network diameter) and $l(x) < u(x)$, it must hold:

$$d \leq \frac{2D(u(x) - l(x))}{H}.$$

and since $H = \frac{3}{4}D$, $d \leq \frac{8}{3}(u(x) - l(x))$. Then, under Assumption 1 we have:

$$\lim_{\Delta \to \infty} d = \frac{8}{3}r$$

■

B. *Existence of bounds on Euclidean distance*

The analysis of the previous subsection relies on Assumption 1 to prove that the distance between nodes in the same zone approaches $\frac{8}{3}r$ as the network density increases.

To validate Assumption 1, we study the relationship between a given hop count and the bounds on Euclidean distance using an approach that minimizes the distance from the next node to the destination. As shown in Figure 6, the current distance between a node A and the destination B is d. P is potential neighbor of A and its distance to B is **z**, which is a random variable. A Greedy approach chooses as next hop the node P that minimizes **z**. We consider the node density to be sufficiently high to avoid any potential backward movement at any step due to network holes.
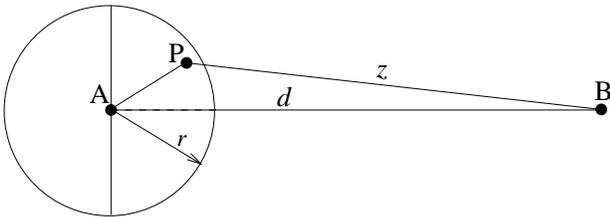

Fig. 6.   Remaining Euclidean distance to the destination from the next hop node in greedy approach.

Given a distance d between the next hop node and the destination, following [28], we have the probability density function of progress in Euclidean distance **x** per hop using the greedy approach $f_{\mathbf{x}}(x)$ is given by Equation (2). Using $f_{\mathbf{x}}(x)$ we obtain the bounds on Euclidean distance for different hop counts by numerical simulations.

A set of representative data are shown in Table I. It is noted that the bound gets tighter with higher node
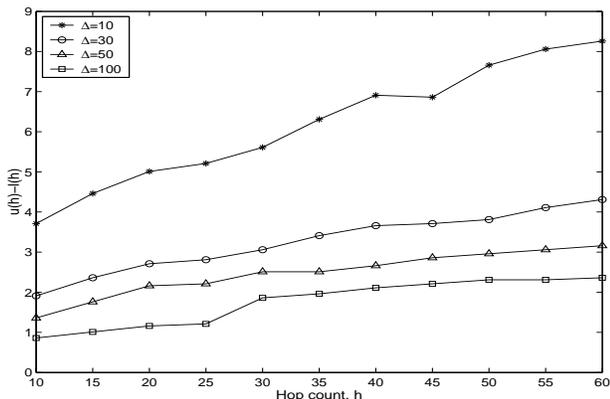

Fig. 7.   Ambiguity in Euclidean distance in unit of nodal range.

TABLE I
BOUNDS ON EUCLIDEAN DISTANCE IN UNIT OF NODAL RANGE
FOR DIFFERENT NODE DENSITIES.

| Hop count | $\Delta = 10$ | | $\Delta = 30$ | | $\Delta = 50$ | |
	$l(h)$	$u(h)$	$l(h)$	$u(h)$	$l(h)$	$u(h)$
20	12.3	16.6	15.5	18.2	16.3	18.7
40	26.5	32.0	32.4	36.1	34.2	37.1
60	40.7	47.4	49.5	53.6	52.1	55.3

density.

Numerical results plotted in Figure 7 show the ambiguity in Euclidean distance for a given hop count. We observe that with higher node density, the ambiguity in distance becomes nearly saturated beyond a certain hop count. Therefore, for all practical purposes, for a given network size and a given node density, we can consider the maximum limit of distance ambiguity $u(h) - l(h)$ as the near-saturation limit.

VI. SIMULATIONS

In this section we present the outcome of our simulation experiments with two aims:

- To study the properties of the virtual coordinate system and compare the simulation outcomes with analytical results. In particular, we study statical properties of the largest *zone*, i.e. the largest set of nodes sharing the same virtual coordinates.
- To evaluate the reachability of nodes using a simple routing algorithm based on virtual location information. We measure the success rate of path construction for random source-destination pairs and obtain the average hop count. These results are compared with the same measures collected using physical coordinates, where the actual location of the nodes are assumed to be known.

A. *Distance of nodes sharing the same coordinates*

These simulations are aimed at the evaluation of the worst case, i.e. the case in which the zones span the largest number of hops.

To this purpose, for a given density, the simulator randomly generates 1000 networks, it assigns the virtual coordinates using VCap and it measures the maximum distance d_N between two nodes with the same coordinate. The simulation evaluates the average $E(d_N)$, the maximum $Max(d_N)$, and the minimum $min(d_N)$.

The simulation results obtained in a circular network of radius 1, transmission range $r = 0.05$, and uniform distribution of nodes are in Figures 8 and 9. Figure 8 shows the values of $E(d_N)/r$, $Max(d_N)/r$, and $min(d_N)/r$ for network densities in the range of 10

$$f_{\mathbf{x}}(x) = \begin{cases} \Delta \left(\frac{2}{\pi r^2}\right)^{\frac{\Delta}{2}} (d-x) \left[\frac{\pi}{2} - \arcsin\left(1 + \frac{x^2-r^2}{2d(d-x)}\right)\right] \left[(d-x)^2 \arcsin\left(1 + \frac{x^2-r^2}{2d(d-x)}\right)\right. \\ \left. + \frac{1}{2}\sqrt{4r^2d^2 - (x^2-r^2-2dx)^2} - r^2 \arcsin\left(\frac{x^2-r^2-2dx}{2dr}\right) - \frac{\pi(d-x)^2}{2}\right]^{\frac{\Delta}{2}-1}, \quad r \geq x \geq 0 \\[4pt] \Delta \left(\frac{2}{\pi r^2}\right)^{\frac{\Delta}{2}} (d-x) \left[\arcsin\left(\frac{d}{d-x}\right) - \arcsin\left(1 + \frac{x^2-r^2}{2d(d-x)}\right)\right] \cdot \\ \left[(d-x)^2 \arcsin\left(1 + \frac{x^2-r^2}{2d(d-x)}\right) + \frac{1}{2}\sqrt{4r^2d^2 - (x^2-r^2-2dx)^2}\right. \\ \left. - r^2 \arcsin\left(\frac{x^2-r^2-2dx}{2dr}\right) - (d-x)^2 \arcsin\left(\frac{d}{d-x}\right) - d\sqrt{x^2-2dx}\right]^{\frac{\Delta}{2}-1}, \quad 0 \geq x \geq d - \sqrt{d^2+r^2} \\[4pt] 0, \quad \text{elsewhere.} \end{cases} \quad (2)$$

Fig. 8. Average distance between points sharing the same coordinates.

(a)

(b)

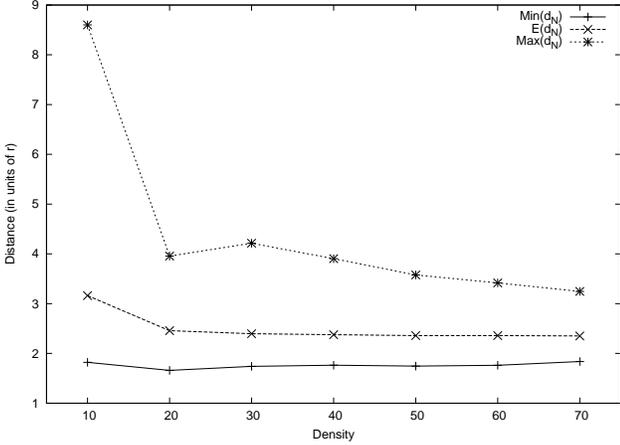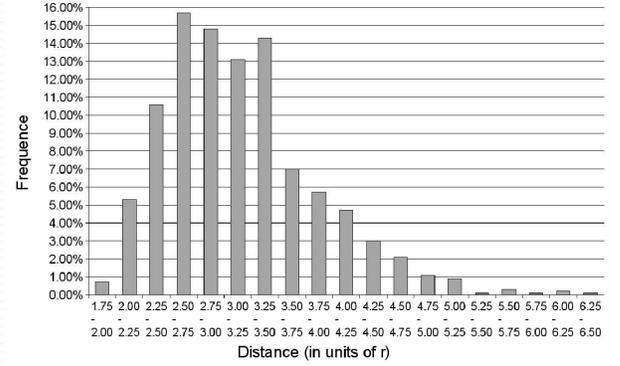Fig. 9. Distribution of distance between points sharing the same coordinates.

to 70, and Figure 9 show the frequency of the different values of d_N grouped in buckets of size $r/4$ for density 10 and 60.
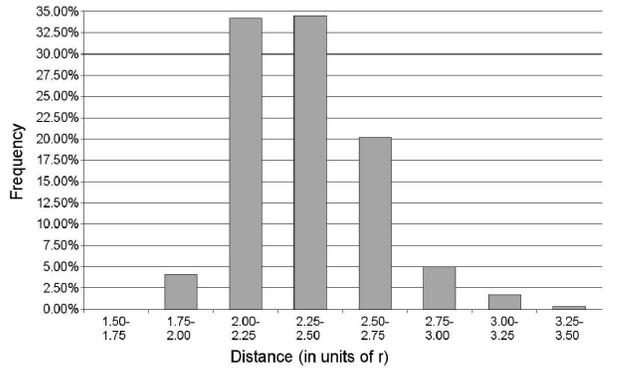
It is seen from Figure 8 that for low density (i.e., 10 neighbors on average) results are quite dispersed, but as density increases, $E(d_N)$ approaches $2.35r$, i.e., it is slightly more than twice the transmission range, while $Max(d_N)$ tends to decrease with the network density and is slightly above $3r$ for density 70. Furthermore, as shown in Figure 9(b), the cases in which $Max(d_N)$ is above $3r$ are rather infrequent. This appears to confirm the result of Theorem 1, which states that for high enough densities the upper bound to $Max(d_N)$ should approach $8r/3$.

B. Performance of geographic routing with virtual co-ordinates

The simulation experiments in this section are aimed at the evaluation of the effectiveness of the virtual coordinate system when used to support a simple geographic routing algorithm. In particular, we study the greedy geographic routing with our hop-based virtual coordinates and compare the network performance with the standard physical coordinate based greedy routing.

Note that our aim is to evaluate the possibility of setting a path, from the source to the destination, using virtual coordinates in very large networks. We do not concern with the problem of actually routing data packets. For this reason our simulation model does not include details of a realistic MAC protocol. In particular, we do not consider neither packet loss nor signal propagation, and we assume that the nodes transmission range is a perfect circle of radius r.

The simulator uses a simple greedy geographic routing which selects as next hop the node for which the distance to the destination is minimum. In the virtual coordinate system, we have defined the distance between two nodes

of coordinate (x_1, y_1, z_1) and (x_2, y_2, z_2) as:

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}.$$

A *local detour* rule is introduced to improve on pure greedy approach by allowing backward progress, if needed. If the routing protocol reaches a node v which is local minimum and it cannot progress toward the destination, the local detour rule allows for a limited number of times the possibility to forward the packet to the node in the neighbourhood of u that is closest to the destination, even if u is farther than v to the destination. This rule improves the success rate of packet delivery but increases the average path length.

In the experiments we considered square domains with side L and circular domanis of radius R. The simulator takes as parameter the side L of the square area (or the radius R of the circular area), the transmission range r of the nodes, the network density Δ expressed as the average number of neighbours per node, and the maximun number c of allowed local detours per packet, and returns the reachability rate, i.e., the probability that the routing algorithm is able to find a path between an arbitrary pair of nodes and the average path lengths with both physical and virtual coordinates.

In each simulation experiment, the simulator generates a random sensor network and executes the VCap protocol to assign the virtual coordinates. Then it applies the routing algorithm to 100 randomly selected source-destination pairs using both virtual and physical coordinates. The simulation is repeated with 10 different seed values to evaluate success rates of routing and average path lengths with the two coordinate systems (physical and virtual). Recall that, with the virtual coordinate system the simulator considers that a node is reachable when it is possible to reach the zone to which the destination belongs.

The simulation results about rechability in square and circular domains are shown in figures 10-11, while the results about path lengths in square and circular domains are shown in figures 12-13.

It is seen that the behaviour of routing in circular networks is slightly better than than in square domains. This is probably due to a border effect which is smaller in the case of the circular location space.

With both virtual and physical coordinates the rechability approaches 100% as the network density increases, however the routing with physical coordinates performs generally better. Routing with virtual coordinates approaches the performance obtained with physical coordinates with $c = 1$ (i.e., the number of allowed local detours per packet). Parameter c appears to have negligible effect on routing with physical coordinates.
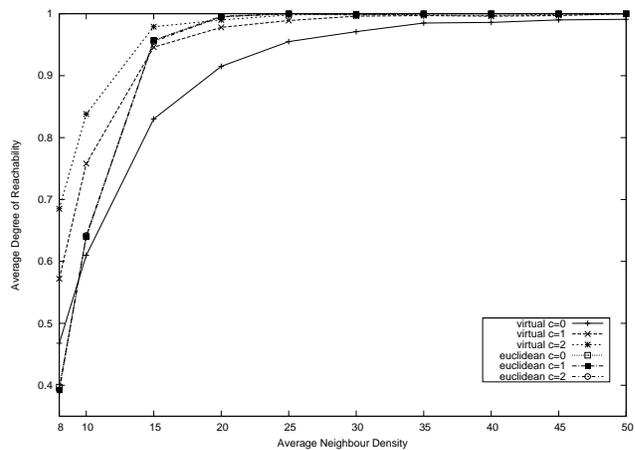

Fig. 10. Packet delivery rates of greedy geographic routing using virtual and physical coordinates in square domains with: $L = 200$, $r = 8$, Δ ranging from 8 to 50, and $c \in \{0, 1, 2\}$.

Fig. 11. Path lenghts of greedy geographic routing using virtual and physical coordinates in square domains with: $L = 200$, $r = 8$, Δ ranging from 8 to 50, and $c \in \{0, 1, 2\}$.

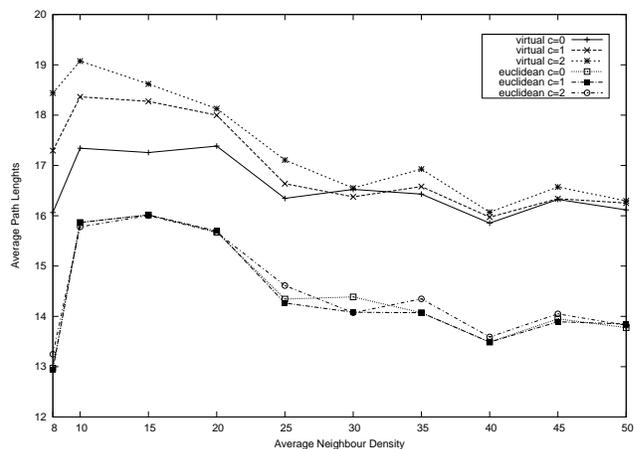However it is interesting to note that, with very low density, routing with virtual coordinates performs better. This is due to the fact that, at low network densities, the virtual coordinates propagate through the available paths, and in some cases, turn around holes in the networks.

The path length with virtual coordinates is around $30\% - 50\%$ larger than with physical coordinates. This is due to the fact that physical coordinates give more information about the relative distance to the destination as compared to virtual coordinates. On the other hand, virtual coordinates do not require GPS equipped nodes.

VII. CONCLUSIONS

In this paper we have considered the problem of constructing a coordinate system in a sensor network where location information is not available. We have introduced the VCap (virtual coordinate assignment) protocol which selects three anchors to define a virtual
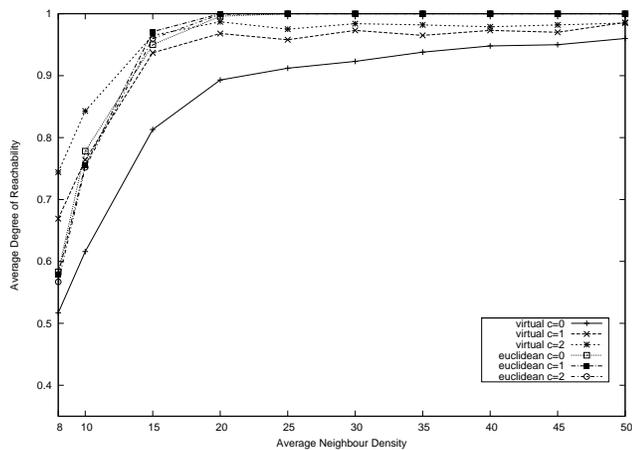
Fig. 12. Packet delivery rates of greedy geographic routing using virtual and physical coordinates in circlular domains with: $R = 100$, $r = 8$, Δ ranging from 8 to 50, and $c \in \{0, 1, 2\}$.

Fig. 13. Path lenghts of greedy geographic routing using virtual and physical coordinates in circlular domains with: $R = 100$, $r = 8$, Δ ranging from 8 to 50, and $c \in \{0, 1, 2\}$.
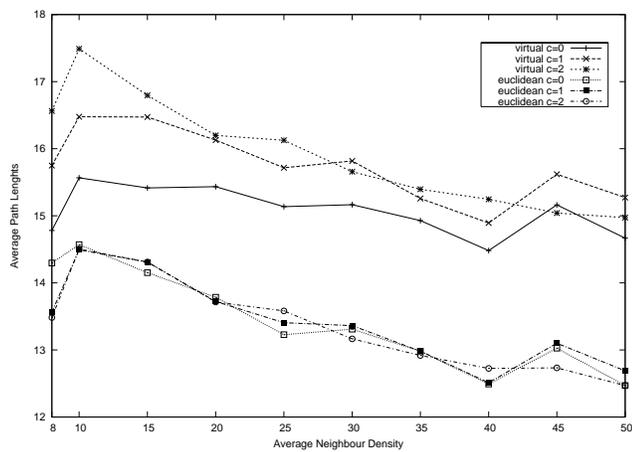
coordinate system based on hop count metric. Compared to other approaches, the VCap protocol is simple and have little requirements in terms of memory overhead.

The virtual coordinate system do not assign individual coordinates to nodes, rather, it defines zones of nodes assigned with the same coordinate. We have shown that the size of the zones is bounded and the upper bound approaches $8r/3$ (where r is the transmission range of the nodes) as the density approaches to infinity. This result have also been confirmed by simulation.

We have evaluated the impact of the virtual coordinate system on routing by simulation, and it is observed that the performance of a simple greedy geographic routing with virtual coordinates is slightly below the routing performance with physical coordinates, but that this difference becomes negligible as the network density increases.

Futures work includes the study of our virtual coordi-nate system using more realistic simulation models and considering mobility. Another area of investigation is the study of routing protocols more suitable to the virtual coordinate system, possibly using two hop information.

REFERENCES

[1] Ian F. Akyildiz, WellJan Su, Yogesh Sankarasubramaniam, and Erdal Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, pp. 102–114, Aug. 2002.

[2] G.N.C. Kirby, A. Dearle, R. Morrison, M. Dunlop, R.C.H. Connor, and P. Nixon, "Active architecture for pervasive contextual services," in *International workshop on middleware for pervasive and ad-hoc computing (mpac 2003), acm/ifip/usenix international middleware conference*, Rio De Janeiro, Brazil, 2003, pp. 21–28.

[3] Deborah Estrin, Ramesh Govindan, John S. Heidemann, and Satish Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of the 5th annual acm/ieee international conference on mobile computing and networking*, Seattle, WA, USA, 1999, pp. 263–270.

[4] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th annual ACM/IEEE international conference on mobile computing and networking*, Boston, MA, USA, 2000, pp. 56–67.

[5] Crossbow Technology, "Mica wireless measurement system datasheet," .

[6] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 131–146, 2002.

[7] Samuel R. Madden and Michael J. Franklin, "Fjording the stream: An architecture for queries over streaming sensor data," in *Icde conference*, San Jose, CA, USA, 2002.

[8] Sylvia Ratnasamy, Brad Karp, Scott Shenker, Deborah Estrin, Ramesh Govindan, Li Yin, and Fang Yu, "Data-centric storage in sensornets with ght, a geographic hash table," *Mobile Networks and Applications (MONET)*, vol. 8, no. 4, pp. 427–442, 2003.

[9] B. Karp and H. T. Kung., "Greedy perimeter stateless routing (gpsr) for wireless networks," in *Proc. sixth annual ACM/IEEE international conference on mobile computing and networking (mobicom)*, Boston, MA, USA, 2000, pp. 243–254.

[10] Wen-Hwa Liao, Jang-Ping Sheu, and Yu-Chee Tseng, "GRID: A fully location-aware routing protocol for mobile ad hoc networks," *Telecommunication Systems*, vol. 18, no. 1-3, pp. 37–60, 2001.

[11] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu, "Geometric spanner for routing in mobile networks," in *Proceedings of the 2nd acm symposium on mobile ad hoc networking & computing (mobihoc'01)*, Long Beach, CA, USA, October 2001, pp. 45–55.

[12] D. Johnson, D. Maltz, and J. Broch, *Ad hoc networking*, chapter DSR The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks, pp. 139–172, Addison-Wesley Longman Publishing Co., Inc., 2001.

[13] Charles E. Perkins and Elizabeth M. Royer, "Ad hoc on-demand distance vector routing," in *Proceedings of the 2nd ieee workshop on mobile computing systems and applications*, New Orleans, LA, USA, February 1999, pp. 90–100.

[14] Jeffrey Hightower and Gaetano Borriello, "Location systems for ubiquitous computing," *IEEE Computer*, vol. 34, no. 8, pp. 57–66, August 2001.

[15] Nissanka B. Priyantha, Allen K.L. Miu, Hari Balakrishnan, and Seth Teller, "The cricket compass for context-aware mobile applications," in *Proceedings of the 7th annual IEEE/ACM international conference on mobile computing and networking (mobicom)*, Rome, Italy, July 2001, pp. 1–14.

[16] Andreas Savvides, Chih-Chieh Han, and Mani B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proceedings of the 7th annual ACM/IEEE international conference on mobile computing and networking (mobicom)*, Rome, Italy, 2001, pp. 166–179.

[17] D. Niculescu and B. Nath, "Ad hoc positioning system (aps) using aoa," in *Proceedings of IEEE infocom*, San Francisco, CA, USA, 2003.

[18] D. Niculescu and Badri Nath, "Error characteristics of ad hoc positioning systems (aps)," in *Proceedings of the 5th acm international symposium on mobile ad hoc networking and computing*, Roppongi Hills, Tokyo, Japan, 2004, pp. 20–30.

[19] Paramvir Bahl and Venkata N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proceedings of IEEE Infocom*, Tel-Aviv, Israel, 2000, pp. 775–784.

[20] D. Niculescu and B. Nath, "Ad hoc positioning system (aps)," in *Proceedings of globecom*, San Antonio, TX, USA, 2001.

[21] Bachrach, Radhika Nagpal, Micheal Salib, and Howard, "Experimental results and theoretical analysis of a self-organizing global coordinate system for ad hoc sensor networks," *Telecommunications Systems Journal, Special Issue on Wireless System Networks*, 2003.

[22] B. Nath D. Niculescu, "Position and orientation in ad hoc network," *Ad Hoc Networks*, vol. 1, no. 2, pp. 133–151, 2004.

[23] Yi Shang, Wheeler Ruml, Ying Zhang, and Markus P. J. Fromherz, "Localization from mere connectivity," in *Proceedings of the 4th acm international symposium on mobile ad hoc networking & computing*, Annapolis, MA, USA, 2003, pp. 201–212.

[24] Ingwer Borg and Patrick Groenen, *Modern multidimensional scaling: theory and applications*, Springer, New York, 1997.

[25] Warren S. Torgerson, "Multidimensional scaling of similarity," *Psychometrika*, vol. 30, pp. 379–393, 1965.

[26] Ananth Rao, Christos Papadimitriou, Scott Shenker, and Ion Stoica, "Geographic routing without location information," in *Proceedings of the 9th annual international conference on mobile computing and networking*, San Diego, CA, USA, 2003, pp. 96–108.

[27] Ioannis G. Tollis, Roberto Tamassia, Giuseppe Di Battista, and Peter Eades, *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall PTR, 1998.

[28] S. De et al., "Bounds on hop count in greedy routing approach in ad hoc multihop wireless networks," *submitted for pubblication*, 2004.