Localization and Routing in Sensor Networks by Local Angle Information

Jehoshua Bruck* Jie Gao† Anxiao (Andrew) Jiang*

ABSTRACT

Location information is very useful in the design of sensor network infrastructures. In this paper, we study the anchor-free 2D localization problem by using local angle measurements in a sensor network. We prove that given a unit disk graph and the angles between adjacent edges, it is NP-hard to find a valid embedding in the plane such that neighboring nodes are within distance 1 from each other and non-neighboring nodes are at least distance 1 away. Despite the negative results, however, one can find a planar spanner of a unit disk graph by using only local angles. The planar spanner can be used to generate a set of virtual coordinates that enable efficient and local routing schemes such as geographical routing or approximate shortest path routing. We also proposed a practical anchor-free embedding scheme by solving a linear program. We show by simulation that not only does it give very good local embedding, i.e., neighboring nodes are close and non-neighboring nodes are far away, but it also gives a quite accurate global view such that geographical routing and approximate shortest path routing on the embedded graph are almost identical to those on the original (true) embedding. The embedding algorithm can be adapted to other models of wireless sensor networks and is robust to measurement noise.

Categories and Subject Descriptors: E.1 [Data]: Data Structures—*graphs and networks*; F.2.2 [Theory of Computation]: analysis of algorithms and problem complexity—*non-numerical algorithms and problems*

General Terms: Algorithms, Design, Theory

Keywords: Sensor networks, Wireless networks, Localization, Geographical routing, Embedding, Planar spanner subgraph

* Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125. Email: {bruck, jax}@paradise.caltech.edu.

† Center for the Mathematics of Information, California Institute of Technology, Pasadena, CA 91125. Email: jgao@ist.caltech.edu.

MobiHoc'05, May 25–27, 2005, Urbana-Champaign, Illinois, USA.

1 Introduction

The fast development of sensor networks in recent years has attracted a lot of interest in the networking community. Sensor networks, with their flexible and scalable nature, have great potential for a variety of applications such as environment monitoring, digital battlefield, etc. Unlike other networks with more logical structures, sensor networks are closely related to the geometric environment where they are deployed. In particular, location information has been proven to be very useful in the design of sensor network infrastructures. First of all, a sensor network is "data centric", where individual sensors are not as interesting as their sensed data. But the data sensed by sensor networks, such as temperature or humidity, are meaningless if we don't know where the data are from. Location information can also help routing. For example, geographical routing makes use of the location of sensor nodes as a routing criterion, where a node sends the message to the neighbor who is closest to the destination. Under dense sensor deployment, this greedy routing will successfully deliver the message to the destination in a local and efficient manner.

Location information can be obtained by using global positioning systems (GPS) [14]. But GPS is expensive and does not work indoor. So there has been a lot of study on localization algorithms that induce the locations of sensor nodes from their local interactions, such as the detection of local neighbors and/or the distances (angles) between neighbors [28, 29, 22, 23, 24, 27, 30, 31, 2, 20, 13, 8]. Many of them assume the existence of a (sometimes large) number of anchor nodes whose positions are already known [28, 29, 22, 23, 30, 24, 8]. In this paper we focus on anchor-free methods that deduct the geometry of the network by only the interactions of the nodes. Existing anchor-free algorithms can be classified into two categories based how much information they use. Some of them use only the connectivity of the communication graph [27, 30]. Others also use the distances between sensor nodes, which can be estimated by Received Signal Strength Indicator (RSSI) or Time of Arrival (ToA) techniques. Distance estimation can be severely affected by the environment and is usually unreliable and noisy [10]. Another problem with distance information is that when the localization solution is not unique, the embedded graph may have incorrect folding, where some pieces of the graph fold on top of other pieces without violating the distance constraints. Finding an embedding without incorrect folding is a challenging research problem.

Interestingly enough, little work has been done on using local angle information for localization. Angles between adjacent edges can be measured by using multiple ultrasound receivers [25], in particular, the Angle of Arrival (AOA), or by using directional antennas. Considering angle information adds one more dimension to the localization problem. Intuitively the angle information tells us how the graph stretches out in different directions and prevents incorrect folding, thus the localization problem could be made easier.

In this paper, we study what can and what cannot be done using the connectivity together with the local angle information. Given a combinatorial unit disk graph G with the angles between adjacent edges specified, we want to find a valid embedding of G in the plane. That is, we want to assign Euclidean coordinates to the vertices of G such that G is the induced unit disk graph that meets the angle constraints. We prove that this problem is hard. Specifically, it's NP-hard to find a valid embedding where neighboring nodes are embedded no further than distance 1 from each other and non-neighboring nodes are embedded at least distance 1 away. And actually it's still hard to find a solution to some relaxed problems. In particular, it's NP-hard to find a $\sqrt{2}$-approximate embedding where non-neighboring nodes are embedded at least $\sqrt{2}/2$ away, or a topologically-equivalent embedding where two edges cross in the embedded graph if and only if they cross in a valid embedding.

In spite of the difficulty of embedding, with local angle information we can find a subgraph G' of G such that for *any* valid embedding $\mathcal{E}$ of G, the graph $\mathcal{E}(G')$ induced by the same embedding is a planar spanner of $\mathcal{E}(G)$. Specifically, no two edges cross in $\mathcal{E}(G')$ and the shortest path distance between two nodes in $\mathcal{E}(G')$ is at most a constant factor of that in $\mathcal{E}(G)$. Spanner subgraphs are very useful in topology control and geographical routing. In particular, geographical routing uses face routing on a planar subgraph to guide a packet out of the local minima. There has been lots of work on constructing planar spanner subgraphs of unit disk graphs (please refer to [26] for an overview), but all of them assume that locations are already known. Here we are the first to show that actually one does not need as much as the location information to construct a planar spanner, only the local angle information suffices! Further, a straight line embedding of the combinatorial graph G' in the plane gives a set of virtual coordinates for sensor nodes with which the geographical routing is guaranteed to deliver a packet to its destination if such a path exists. This shows that just for the purpose of geographical routing, using accurate location information is unnecessary.

For practical applications, we propose an embedding algorithm with local angle information that gives surprisingly good results. We first formulate the embedding problem by a linear program with relaxed constraints such that any valid embedding must be a feasible solution to the LP. Through simulations, we show that the LP finds an almost identical set of locations as the original ones, even when the graph is sparse. We also show that the method is robust to both noisy measurements of angles and different models of sensor networks — specifically, the more general quasi-unit disk graph models. A planar spanner derived based on local angle information equipped with the virtual coordinates obtained through embedding enables geographical routing and approximate shortest path routing with demonstrated performance almost the same as using the real locations.

2 Related work

2.1 Localization

The localization problem, i.e., determining the global geometry by using only local information, was studied in many communities such as computational biology, machine learning, and sensor networks. The localization problem can be formulated as a graph embedding problem, i.e., to embed the vertices of a graph in a geometric space such that the embedded drawing satisfies desired properties. A sensor network is modelled by a *unit disk graph (UDG)*, where two nodes are connected by a communication link if and only if their Euclidean distance is no more than 1. Unit-disk graph embedding is to find an embedding of the vertices in the Euclidean plane such that the distance between two nodes is at most 1 if there is an edge, and the distance between two nodes is more than 1 if they don't have an edge.

By using purely the connectivity information, it's known that determining whether a combinatorial graph is a unit-disk graph is NP-hard, and thus finding such an embedding is also hard [5]. In fact, even a relaxed version of the problem is still hard. It's shown by Kuhn *et al.* that finding an embedding such that non-neighboring pairs are at least 1 away and neighboring pairs are within $\sqrt{3/2}$ is NP-hard [16]. There have been a number of heuristics proposed for localization by mere connectivity [27, 30]. But not much is known on the worst case bound. So far the only known theoretical result is an algorithm with an upper bound $O(\log^{2.5} n\sqrt{\log\log n})$ on the ratio of the longest distance between neighboring pairs to the shortest distance between non-neighboring pairs [21]. In the localization problem with range information, we are also given the distances between certain node pairs in the graph besides the connectivity. If only edge lengths are provided, finding a feasible embedding is NP-hard [1, 7] (also proved by a slight variation of the proof in [5]). When all pairs of inter sensor distances are known, the solution is unique and can be solved by the classical Multidimensional Scaling (MDS) method [3]. If the distances between about $\Omega(n^2)$ pairs of nodes are given and there is a unique solution, the embedding problem can be formulated as a semi-definite program and solved in polynomial time [31, 2].

In practice, many localization algorithms assume the existence of anchor nodes whose locations are known by GPS or other methods. Then trilateration is used to find the locations of the sensors progressively [28, 29, 22, 20, 8]. If the distances from a sensor p to three anchors are known, the location of p is uniquely determined. Similar methods can also be done by using angles [23, 24]. These incremental solutions usually suffer from cascading errors and the localization result can be beyond tolerable on large-scale networks.

2.2 Geographical routing

Geographical routing is a local and efficient routing algorithm proposed for ad hoc networks. In the traditional geographical routing, each node knows its own location. A source node knows the location of the destination and uses it as the goal of routing. Geographical routing is composed of two schemes, greedy forwarding and perimeter routing (also called face routing) [15, 4]. In greedy forwarding, a message is forwarded to the neighbor whose Euclidean distance to the destination is the minimum among all neighbors. When a message gets stuck at a node whose neighbors are all further away from the destination, it uses perimeter routing to route along the faces of a planar subgraph until either the destination is reached or greedy forwarding can be performed again. Perimeter routing can also be improved by an 'early-fallback' technique to return to greedy routing as soon as possible [17]. In all these variations, both the location information and a correctly constructed planar subgraph are needed.

Due to the hardness of the localization problem, people have proposed various schemes of computing virtual coordinates in replace of the real coordinates. The most prominent work is done by Rao *et al.* [27], where they construct a set of virtual coordinates by using only the connectivity for geographical routing. But when a message gets stuck at a local minima, the only way for it to reach the destination is to be flooded to the whole network. Comparably, we use more information, the local angle information, and produce an embedded planar spanner subgraph together with a set of virtual coordinates such that stuck messages can be routed to the destination by perimeter routing.

3 Preliminaries

We start with some definitions on unit disk graphs and embeddings. Throughout the paper we assume that the UDG is connected since otherwise we'll work on each connected component separately.

Definition 3.1. *A* unit-disk graph *is a combinatorial (unweighted) graph induced by a set of points in the Euclidean plane such that two points have an edge in between if and only if their distance is no more than 1.*

We emphasize here that by the notion of unit-disk graph we mean the combinatorial graph without the embedding. Such a unit-disk graph is induced by a set of points in the Euclidean plane but the configuration of the nodes in $\mathbb{R}^2$ is unknown. An embedding of such a combinatorial graph in the Euclidean plane may or may not be the same as the original (unknown) configuration. For an embedding $\mathcal{E}$, we denote by $\mathcal{E}(p)$ the embedded point of a node p. The Euclidean distance between two nodes p, q in an embedding $\mathcal{E}$ is denoted by $d(\mathcal{E}(p), \mathcal{E}(q))$. We will sometimes abuse the notations and use p to represent $\mathcal{E}(p)$ when the context is clear.

In this paper we study embedding problems by using local angle information. Specifically, besides the combinatorial unit disk graph we are also given the angles between angularly adjacent edges (All angles are measured counterclockwise). See Figure 1. With the local angles constrained there is still freedom to choose the lengths of the edges.

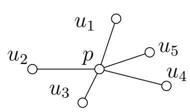

Figure 1. For each node p in the unit disk graph, assume that $u_1, \cdots, u_k$ are p's neighbors ordered counterclockwise. In this paper we assume that the angles between edges pu_i and pu_{i+1} are given.

Definition 3.2. *An* α-approximate embedding $\mathcal{E}$ *of a graph* G *with angle information is an embedding of the vertices such that the distance between two nodes $d(\mathcal{E}(u), \mathcal{E}(v)) \le 1$ if u, v have an edge between them in G, and $d(\mathcal{E}(u), \mathcal{E}(v)) > 1/\alpha$ if u, v don't have an edge between them in G, where $\alpha \ge 1$. The angle between any two adjacent edges uv, uw is as specified.* A valid embedding *is an α-approximate embedding with $\alpha = 1$.*

We observe that by local angle information, we can decide whether two edges cross in a valid embedding of the unit disk graph. Thus when we say two edges cross in a unit disk graph G, we actually mean that they cross in any valid embedding of G.

Lemma 3.3. *If we know the angles between adjacent edges of a unit disk graph, we can determine all pairs of crossing edges in a valid embedding.*

PROOF. In particular, if two edges AB, CD intersect with each other, there must be a node that is connected with all the other three nodes [5, 11]. Suppose B is connected with the other three nodes. Then AB, CD cross each other if and only if AB is located inside the cone defined by $\angle CBD < \pi$ and A, B are on different sides of the line defined by CD.

First we can decide if AB is located inside the cone defined by $\angle CBD < \pi$ easily by the angle information. Further, if AB is located inside the cone defined by $\angle CBD$ and A, B are on the same side of the line defined by CD, then A is inside the triangle BCD. See Figure 2 (ii). Then A is connected to B, C, D due to plane geometry. This situation can be identified since BA must be outside the cone defined by $\angle CAD$. $\square$

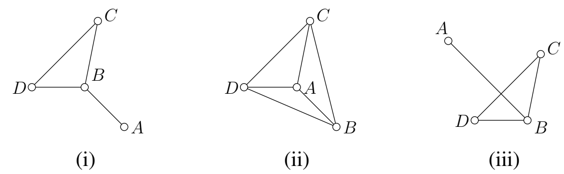

Figure 2. (i) The edge AB is not located inside the angle $\angle CBD$ and thus AB, CD cannot cross each other; (ii) AB is located inside the cone defined by $\angle CBD$ and A, B are on the same side of the line defined by CD, then BA must be outside the cone defined by $\angle CAD$; (ii) A correct crossing between AB and CD.

The above lemma implies that we can identify all crossing edges in a valid embedding with local angle information. Thus one relaxation of a valid embedding is to require that the topology of the embedded graph is equivalent with a valid embedding, i.e., only the edges that cross in a valid embedding are allowed to cross.

Definition 3.4. *A* topologically equivalent embedding $\mathcal{E}$ *of a graph* G *with angle information is an embedding of the vertices such that two edges cross in $\mathcal{E}$ if and only if they cross in a valid embedding. The angle between any two adjacent edges uv, uw is as specified.*

Remark. We notice that without loss of generality we can assume that in a topologically equivalent embedding the neighboring nodes are embedded no further than distance 1. This is because we can always do proper global scaling that doesn't change the topology of the embedded graph.

Theorem 3.5. *A $\sqrt{2}$-approximate embedding is a topologically equivalent embedding.*

PROOF. Assume that there are two edges AB, CD that cross each other in a $\sqrt{2}$-approximate embedding $\mathcal{E}$. Also assume that $\mathcal{E}^*$ is a valid embedding. If the following two claims are true, then $\mathcal{E}$ is topologically equivalent with $\mathcal{E}^*$.

Claim 1: If AB, CD cross in a valid embedding $\mathcal{E}^*$, then they must also cross in a $\sqrt{2}$-approximate embedding $\mathcal{E}$.

Proof of claim 1. If AB, CD cross in a valid embedding $\mathcal{E}^*$, then one node must be connected to all the three other nodes. There are three possible cases, as illustrated by Figure 3. For case (ii) and (iii), if the angles between adjacent edges are fixed as specified, the configuration of the four nodes is unique up to a global rigid motion and scaling. Thus AB, CD cross in any embedding preserving the local angles. For case (i), we argue that in a $\sqrt{2}$-approximate embedding AB, CD must also cross each other. In a valid embedding $\mathcal{E}^*$ as in Figure 4 (i), AC must be longer than both AD and CD. Thus the angle $\angle CDA > \pi/3$. Similarly $\angle BDC > \pi/3$. Thus $\angle BDA > 2\pi/3$. If in a $\sqrt{2}$-approximate embedding $\mathcal{E}$, AB doesn't cross CD, then C is embedded inside the triangle ADB, as shown in Figure 4 (ii). First $\angle BCA > \angle BDA > 2\pi/3$. On the

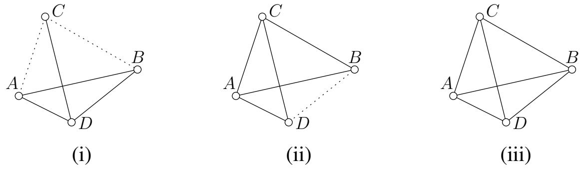

Figure 3. In a valid embedding of the unit disk graph G, if two edges AB, CD cross each other, there are only three possible cases.

other hand, $d(\mathcal{E}(A), \mathcal{E}(C)) > \sqrt{2}/2$, $d(\mathcal{E}(B), \mathcal{E}(C)) > \sqrt{2}/2$, $d(\mathcal{E}(A), \mathcal{E}(B)) \leq 1$. Thus,

$$d(\mathcal{E}(A), \mathcal{E}(C))^2 + d(\mathcal{E}(B), \mathcal{E}(C))^2 > 1 \geq d(\mathcal{E}(A), \mathcal{E}(B))^2.$$

So $\angle BCA < \pi/2$. This leads to a contradiction.

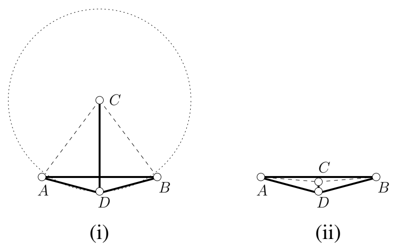

Figure 4. (i) A valid embedding $\mathcal{E}^*$; (ii) A $\sqrt{2}$-approximate embedding $\mathcal{E}$.

Claim 2: If AB, CD cross in a $\sqrt{2}$-approximate embedding $\mathcal{E}$, then they must also cross in a valid embedding $\mathcal{E}^*$.

Proof of claim 2. There are six possible cases based on how the nodes are connected with each other in G. See Figure 5. We argue that none of the cases have both properties that

- AB, CD intersect each other in $\mathcal{E}$ and
- AB, CD don't intersect each other in $\mathcal{E}^*$.

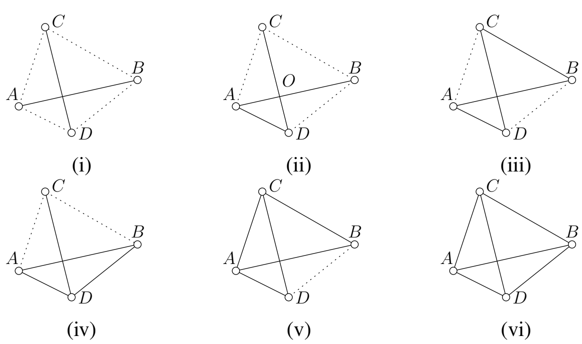

Figure 5. A $\sqrt{2}$-approximate embedding $\mathcal{E}$. Solid lines are edges in G.

1. For case (i) in Figure 5, let's take a look at triangle $\triangle ACD$ under embedding $\mathcal{E}$. We know that $d(\mathcal{E}(A), \mathcal{E}(C)) > \sqrt{2}/2$, $d(\mathcal{E}(A), \mathcal{E}(D)) > \sqrt{2}/2$, $d(\mathcal{E}(C), \mathcal{E}(D)) < 1$. So the angle $\angle CAD < \pi/2$. Similarly, $\angle ACB < \pi/2$, $\angle CBD < \pi/2$, $\angle BDA < \pi/2$. This leads to a contradiction since the sum of the inner angles of a 4-gon must be 2π. So this case can never happen in $\mathcal{E}$.
2. Case (ii) cannot happen for a $\sqrt{2}$-approximate embedding $\mathcal{E}$. The intuition is that if the two edges don't cross in a valid embedding, then the angle $\angle COB \leq \pi/6$. This contradicts with the fact that $d(\mathcal{E}(B), \mathcal{E}(C)) > \sqrt{2}/2$. The details are in Appendix 8.
3. Case (iii) cannot happen. By the angle constraint, the two edges AB, CD must cross in any planar embedding. But in a valid embedding there must be a node that is connected to three other nodes. This leads to a contradiction.
4. For cases (iv), (v) and (vi), AB and CD cross in any valid embedding.

Therefore if two edges don't cross in a valid embedding, they cannot cross each other in any $\sqrt{2}$-approximate embedding. This shows that an $\sqrt{2}$-approximate embedding is a topologically equivalent embedding. $\square$

4 The hardness of UDG embedding with angles

As shown in the last section, by using local angle information we can decide on all crossing edges in a valid embedding. However, local angle information is not sufficient for us to decide a valid embedding. It turns out that the problem of finding a valid embedding by using the connectivity and the local angle information is still hard. In fact it's even NP-hard to find a topologically equivalent embedding or a $\sqrt{2}$-approximate embedding. In this section we show a reduction from the 3SAT problem.

A 3SAT problem consists of a set of Boolean variables and clauses such that each clause is composed of at most 3 literals, which are either negated or unnegated. The 3SAT problem is to find an assignment to the variables such that all the clauses are satisfied. A 3SAT instance C can be formulated as a graph G_C where vertices are the set of clauses and variables, and there is a path connecting a clause with a variable (or its negated version) if the variable appears in the clause. Please see Figure 6 for an example. Such a graph can be drawn on a grid in polynomial time [5]. Breu and

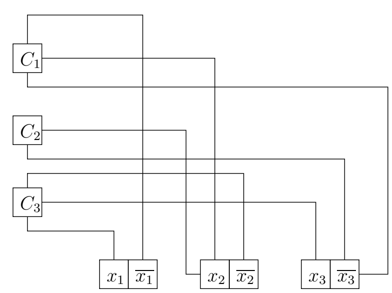

Figure 6. The graph G_C of a 3SAT instance $(\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee x_3)$.

Kirkpatrick proved the NP-hardness of unit disk graph embedding by a reduction from a 3SAT problem [5]. Now we focus on realizing the graph G_C by a unit-disk graph with the angle constraint such that there is a topologically equivalent embedding if and only if the corresponding 3SAT problem is satisfied.

4.1 Basic building blocks

We first present a set of building blocks by using unit disk graphs.

- **Spring**. A spring is a line segment with length between ℓ and 2ℓ. It can be realized by a set of $2\ell + 1$ nodes placed on a straight line such that there are only edges between adjacent pairs, as shown in Figure 7 (ii). In particular, each edge in a unit disk graph has length at most 1, so a chain of $2\ell + 1$ nodes have length at most 2ℓ. For 3 adjacent nodes a, b, c, since a cannot communicate with c, their distance must be at least 1 away. Thus the chain is no shorter than ℓ.

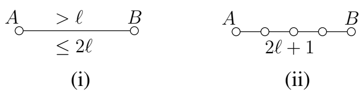

Figure 7. (i) A spring; (ii) The realization of a spring by unit-disk graphs.

- **Amplifier**. An amplifier is a triangle with fixed inner angles. Thus the ratio between the edge lengths of the triangle is fixed. For a number ℓ we can use an amplifier to get the number $\ell' = c \cdot \ell$ for any $c > 0$. An amplifier can be realized by a unit disk graph with pre-specified angles between adjacent edges.

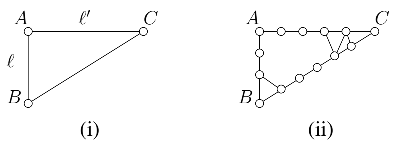

Figure 8. (i) An amplifier; (ii) The realization of an amplifier by unit-disk graphs.

- **Propagator and Crossing Propagator**. A propagator is a rectangle. The lengths of the opposing sides of the rectangle are the same. It can be implemented by a cycle of nodes with corresponding angle constraints. A crossing propagator is a pair of crossing rectangles. See Fig. 9.

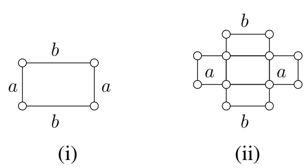

Figure 9. (i) Propagator; (ii) Crossing propagator.

- **0/1 block** By using the above building blocks, we can construct a 0/1 block that has only two types of valid embedding. In short, we construct a concave cycle with one top "tooth" and one bottom "tooth". If we don't allow the teeth to overlap, there are basically two ways to embed the concave cycle, either by putting the top tooth to the left of the bottom tooth, or the other way around. Please see Figure 10 for the two types of embedding. The concave cycle is bounded by $AEFGHDCKLIJB$, the top tooth is the part of the cycle $EFGH$, the bottom tooth is the part of the cycle $JILK$. Suppose the length of $AB = CD$ is ℓ, we use amplifiers and propagators such that the length of $BC = DA = 11\ell/6$. There are two squares $EFGH$, $IJKL$ inside the rectangle $ABCD$. Both of them have side length $2\ell/3$. The two squares don't have edges in between. Thus any embedding without incorrect crossings will have to embed the graph in two ways, either by putting the square $EFGH$ to the left of $IJKL$ or the other way around. In the first case, the length of the path AE is no more than $\ell/2$, the length of HD is at least $2\ell/3$. In the second case, the length of the path AE is at least $2\ell/3$, and the length of HD is no more than $\ell/2$. The segments AE, HD, BJ, KC are springs, thus their lengths can be stretched and shrinked by a factor no more than 2.

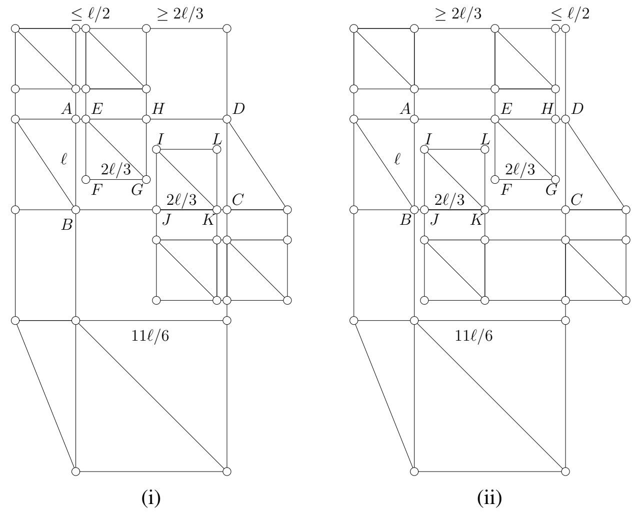

Figure 10. The only two embedding of a concave cycle without incorrect crossings.

4.2 Realization of G_C by unit disk graphs

Now we are ready to introduce how to realize the graph G_C for a 3SAT instance C by using unit disk graphs with angle constraints. The graph G_C consists of three components: clauses, variables and wires to connect them.

- **Wires** The wires are simply propagators. If the width of a propagator is no more than $\ell/2$, this means the variable connected by the wire is assigned '1'. If the width of a propagator is at least $2\ell/3$, the variable connected by the wire is assigned '0' in G_C.
- **Variable components** A variable is implemented by a 0/1 block. In fact, we use the length of AE to represent the value of a variable and the length of HD to represent its negated version. A variable v is assigned 1 if the length of AE is less than $\ell/2$, and 0 if the length of AE is at least $2\ell/3$. Correspondingly we use the length of HD to represent the negated variable $\overline{v}$.
- **Clause components** A clause component puts constraints on the input variables. In particular, it put a total maximum length on the concatenation of springs whose lengths represent the assignments of input variables. See Figure 11 (i) for an example. If a clause is composed of three variables, then the outer rectangle has width $11\ell/6$. Thus at least one of the variable has length less than $\ell/2$. That is, the clause is satisfied if at least one variable is assigned value 1. The clauses with two or one variables are designed similarly. See Figure 11 (ii) and (iii).

Now we put all the components together and show a realization of the graph G_C (Figure 6) for a 3SAT instance C by a unit disk

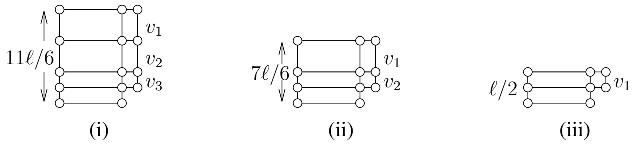

Figure 11. Clause components (i) $(v_1 \vee v_2 \vee v_3)$; (ii) $(v_1 \vee v_2)$; (iii) v_1.

graph in Figure 12. Intuitively, the hardness of the problem is due to that the ways to embed the 0/1 blocks are affected by each other through the constraints put by the clauses.

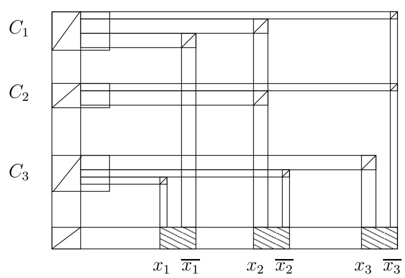

Figure 12. The realization of a 3SAT instance $(\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee x_3)$ by a unit disk graph. Shaded areas are 0/1 blocks for variables. In this example $x_1 = 1$, $x_2 = 0$, $x_3 = 0$. The instance is satisfied.

4.3 Hardness results

Now we are ready to prove the NP-hardness of unit disk graph embedding with local angle information.

Theorem 4.1. *It's NP-hard to find a topologically equivalent embedding of a unit-disk graph with local angle constraints.*

PROOF. By the construction of G_C for a 3SAT instance G_C, we can see that the the instance C can be satisfied if and only if we can find an embedding of G_C in the plane that has the same topology and preserves all the local angles. Since 3SAT is NP-hard, it's also NP-hard to find a topologically equivalent embedding. □

Corollary 4.2. *It's NP-hard to find a valid embedding of a unit-disk graph with local angle constraint.*

PROOF. The proof is similar with the above theorem. For a graph G_C of a satisfiable 3SAT instance C, we can find an embedding $\mathcal{E}$ of G_C with no incorrect crossings. Further we can do proper scaling and local arrangement of $\mathcal{E}$ such that $\mathcal{E}$ is a valid embedding. □

Corollary 4.3. *It's NP-hard to find an α-approximate embedding of a unit-disk graph with local angle constraints, for $\alpha < \sqrt{2}$.*

PROOF. We construct a graph G_C for a 3SAT instance C. By Theorem 3.5, a $\sqrt{2}$-approximate embedding is a topologically equivalent embedding. Thus if we have a $\sqrt{2}$-approximate embedding $\mathcal{E}$ of G_C, then C is satisfiable. The other direction can be proved similarly as the above proof. □

Input	Hardness	ref.
UDG graph only	NP-hard	[5, 16]
$O(1)$-hop distances	NP-hard	[1]
$O(1)$-hop angles	NP-hard	this paper
$O(1)$-hop angles & distances	in P	this paper
$\Omega(n^2)$ pairs distances	in P	[2, 31]
all pairs angles	in P	this paper

Figure 13. A summary of the hardness of finding a valid embedding of a UDG.

4.4 A summary of hardness of localization

Localization by using only angles between adjacent edges in a unit disk graph is shown to be NP-hard. However, if we have more information, localization can be solved easily from a theoretical point of view. For example, if we have the angles between all pairs of nodes in the graph, then the graph is basically determined up to a scaling factor. For another example, if we have both the lengths of the edges and the angles between adjacent edges in a unit disk graph, the graph is uniquely determined. A short summary of the hardness results on localization is shown in Figure 13.

5 Planar spanner construction

In the previous section we've shown that by using the communication graph and local angle information, it's NP-hard to find a valid embedding of a unit disk graph. On the positive side we'll show that by local angle information we can find a planar spanner subgraph whose embedding in the plane can be used for geographical routing with guaranteed delivery.

A planar graph is a graph that can be embedded in the plane with no edge crossings. A c-spanner G' of a graph G is a subgraph of G such that the shortest path distance of u, v in G' is at most c times the shortest path distance of u, v in G, where the shortest path distance is the sum of the Euclidean length of all the edges on the shortest path. c is the spanning ratio of G'. A spanner with a constant spanning ratio is usually called a spanner. In this section we'll show that one can construct a planar spanner for a unit disk graph by using only the angles between adjacent edges. Recall that the location information is not available. Thus when we say a planar spanner we mean a subgraph G' of the input unit disk graph G such that for *any* valid embedding $\mathcal{E}(G)$, the subgraph G' on the same embedding $\mathcal{E}(G')$ is a planar spanner. Finding a spanner subgraph can be easily done without the location information, however, finding a spanner subgraph that has a planar embedding for any valid embedding of the UDG doesn't seem to be intuitive. The idea is to find a planar subgraph that is guaranteed to contain a restricted Delaunay graph, i.e., a subgraph of the Delaunay triangulation with all the edges longer than 1 deleted [11].

A Delaunay triangulation on a point set in $\mathbb{R}^2$ is a triangulation with "empty-circle" property: the circumcircle of any triangle has no other points inside. A restricted Delaunay graph, defined as the subgraph of the Delaunay triangulation with all the edges longer than 1 deleted, is known to be a 2.42-spanner of the unit disk graph [11, 19]. Now we claim that with local angle information we can find a subgraph G' of G that is planar and contains all the edges of a restricted Delaunay graph. Thus G' is a planar spanner subgraph of G with spanning ratio 2.42.

Suppose two edges AB, CD cross each other in a unit disk graph, then only one of them can possibly be a Delaunay edge due to the planar property. We show that we can decide which one is *not* a Delaunay edge by using the local angle information. To be specific, there are only three possible cases of a pair of crossing edges, as shown in Figure 3. Notice that in cases (ii) and (iii), because of the given angle information, the positions of the four nodes are unique up to a rigid motion and a scaling factor. Since the Delaunay triangulation is invariant under global scaling, there is only one possible Delaunay triangulation, which can be decided by only the angles.

For case (i), node C is at least of distance 1 away from nodes A, B. See Figure 14. We take the bisectors of the edge AD, BD, ℓ_1, ℓ_2, that intersect at a point O. O is also the center of the circumcircle of $\triangle ABD$. The lines ℓ_1, ℓ_2 divide the plane into four quadrants. Node C must be inside the same quadrant with node D since $d(\mathcal{E}(C), \mathcal{E}(D)) \leq 1 < d(\mathcal{E}(C), \mathcal{E}(A))$, $d(\mathcal{E}(C), \mathcal{E}(D)) \leq 1 < d(\mathcal{E}(C), \mathcal{E}(B))$. Thus C is inside the circumcircle of $\triangle ABD$. This implies that the edge AB is not a Delaunay edge, since it violates the "empty-circle" property of the Delaunay triangulation.

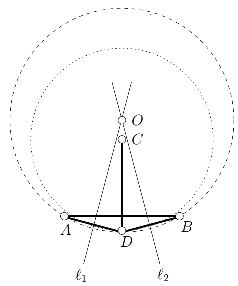

Figure 14. Thick lines are edges in the unit disk graph. Node C must lie in the circumcircle of triangle $\triangle ABD$.

By the above argument, one can decide a non-Delaunay edge between a pair of crossing edges in a unit disk graph. Thus we can eliminate crossings by always deleting non-Delaunay edges. In the end we'll have a planar subgraph G' such that all the Delaunay edges with length no more than 1 are kept. That is, G' contains the restricted Delaunay graph, which is a constant spanner.

Theorem 5.1. *Given a unit disk graph and the angles between adjacent edges, one can construct a planar spanner subgraph with spanning ratio* 2.42.

We should also notice that there are possibly infinitely many valid embeddings of a particular unit disk graph that satisfies the angle constraints. However, the planar spanner we found is the same for all such embedded graphs. This is a little counter-intuitive since Delaunay triangulation has been considered to be very delicate – a tiny movement of a single point can possibly change the whole graph structure. Yet we show that the restricted Delaunay graph has some kind of robustness. Further, such a planar spanner subgraph can help us with efficient routing in a sensor network. In particular, it can be used to produce a set of virtual coordinates for efficient geographical routing, or a set of distributed labels for approximate shortest path routing.

5.1 Geographical routing with guaranteed delivery

It's known that any planar graph has a straight line realization in the plane [9, 6]. By using a straight line embedding of the planar subgraph G', each node is assigned an Euclidean coordinate that can be used in geographical routing [15, 4]. Although in our case the location information can not be obtained unless $P = NP$, the embedded planar subgraph provides a set of virtual coordinates that are equally good for geographical routing. The virtual coordinates guarantee the delivery of a packet if possible at all.

5.2 Approximate shortest path routing

In general, graph labelling is to assign a set of distributed labels to the vertices such that the shortest path can be inferred by using only the labels of the source and destination. In particular, one can compute a set of labels, each with size at most $O(\sqrt{n}\log n)$, on the vertices of a planar graph with n vertices, due to the fact that a planar graph enjoys a $O(\sqrt{n})$ balanced separator [12]. The basic idea is to partition the graph recursively into pieces by small-size separators. The number of recursions is $\log n$. For a separator of a subgraph P, we compute and store distributedly the shortest path trees of P centered at all nodes of the separator. Each node has a label with size $O(\sqrt{n}\log n)$. Therefore with the planar spanner G' of the unit disk graph, we can use the above graph labelling algorithm to construct a set of labels with size $O(\sqrt{n}\log n)$ such that one can find a 2.42-approximate shortest path of G by using only the labels of the source and the destination.

6 A practical solution to UDG embedding and routing with angles

Embedding a unit-disk graph is NP-hard, and it is so even when the restriction is relaxed to be finding a topologically equivalent embedding. In practice, however, we still hope to use the local angle information to find localization that well approximates the true sensor network. The planar spanner of a sensor network is certainly very useful for geographical routing and approximate shortest path routing; yet before the routing works, the spanner firstly needs to be realized in the plane where edges are embedded as straight-line segments not crossing each other. There are currently known straight-line embedding algorithms for planar graphs [9, 6]; however, when such algorithms are applied to planar spanners of UDG, they distort the edge lengths and the relative positions among nodes extremely severely, and thus are not effective in practice. In this section, we show that we can construct an embedding method based on linear programming, which produces very good localization solutions; the solutions lead to nearly optimal routing performance as well; we also demonstrate the robustness of the embedding method to noisy measurements of angles and to more general topological models of sensor networks. This shows that using local angle information to do localization and routing is practically good for sensor networks.

6.1 UDG embedding based on LP

We formulate the embedding problem by solving a linear program. We include as many constraints as possible such that the optimization remains a LP. We take the length of each edge e, $\ell(e)$, as a variable. We arbitrarily pick an edge and make the x-axis be parallel to it. By the fact that we know the angle between any two adjacent edges, the *absolute angle* of every edge e — the counterclockwise angle between the positive x-axis and e — can be uniquely determined. We see every edge as the superposition of two directed edges of opposite directions, whose absolute angles differ by π. Then a valid UDG embedding satisfies the following constraints.

- **Edge-length constraint**. $\forall$ edge e, we have

$$0 \leq \ell(e) \leq 1. \tag{1}$$

- **Cycle constraint**. For any cycle that consists of edges $\{e_1, e_2, \cdots, e_p\}$, where for $1 \leq i \leq p$, the absolute angle of e_i is θ_{e_i}, there exist two constraints

$$\sum_{i=1}^{p} \ell(e_i) \cos \theta_{e_i} = 0, \tag{2}$$

$$\sum_{i=1}^{p} \ell(e_i) \sin \theta_{e_i} = 0. \tag{3}$$

- **Non-adjacent node pair constraint**. For any two adjacent edges e_1, e_2 whose three endpoints do not induce a triangle subgraph, we have

$$\ell(e_1) + \ell(e_2) > 1. \tag{4}$$

- **Crossing-edge constraint**. For any two edges AB and CD crossing each other, one of the four nodes must be connected to all the other three. Let's say D is connected to A, B and C, and AB crosses CD at the point x (see Fig. 15(i)). Then there exists the constraint

$$\ell(CD) \geq |xD| = \ell(AD) \frac{\sin \angle DAB}{\sin(\angle ADC + \angle DAB)}. \tag{5}$$

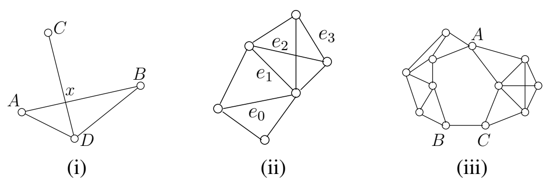

Figure 15. (i) Crossing-edge constraint. (ii) A subgraph where any two edges are related through a sequence of triangles. (iii) Two rigid subgraphs sharing node A and connected by edge BC.

The above constraints serve as the linear constraints in our linear programming. A feasible solution to the LP gives us an embedding of the UDG, since we can use the edge lengths of a spanning tree and the angle information to determine the node positions. There are many ways to select the objective function; as a heuristic, we choose it to be maximizing the minimum length of all edges.

When the UDG has lots of edges, the large number of variables and constraints in the LP will lead to high complexity. In such cases, we can almost always use the following method to significantly reduce the complexity. First we reduce the number of variables. For any three edges AB, BC and CA that form a triangle, since the values of $\angle ABC$, $\angle BCA$ and $\angle CAB$ are given, the three edge lengths have fixed ratios. So we can regard only $\ell(AB)$ as a variable, and represent the lengths of BC and CA respectively by $c_1 \cdot \ell(AB)$ and $c_2 \cdot \ell(AB)$, for some constants c_1 and c_2. Thus three variables are reduced to one variable. Similarly, if a subgraph of the UDG satisfies the condition that for any two of its edges e_0 and e_p, there exist edges $e_1, e_2, \cdots, e_{p-1}$ such that e_{i-1} and e_i are contained in a triangle for $1 \leq i \leq p$ (see Fig. 15(ii) for an example), then all the edge lengths in this subgraph have fixed ratios — therefore they can be represented with only one variable. We call such a subgraph a *rigid subgraph*. To push this approach further, we observe that if several rigid subgraphs share common nodes or are connected by edges, then every cycle that travels through multiple rigid subgraphs enables us to derive two equations like the *cycle constraint* described before. If there are enough such equations, the ratios among the sizes of those subgraphs and the lengths of the connecting edges can be uniquely determined — then those subgraphs and the edges between them unite and form a larger rigid subgraph, all of whose edge lengths can be represented with only one variable. (For example, see Fig. 15(iii), where two rigid subgraphs share the node A and are also connected by an edge BC. All the edge lengths there have determined ratios between themselves and therefore can be represented with only one variable.) The improvement by this approach is large. For example, when 1000 nodes are placed in a 18×18 square with a uniform distribution, the largest connected component typically contains more than 4500 edges; by the above approach, the number of variables in the LP can nearly always be reduced to be less than 30. Then the number of linear constraints can also be reduced.

The above method not only reduces complexity, but also gives us additional constraints for further guarantee on the quality of the embedding. For any two non-adjacent nodes A and B in a rigid subgraph, let $\ell(e)$ denote the edge length in the subgraph specially chosen to be the variable, then $|AB| = c \cdot \ell(e)$ for some constant c. We include the constraint $c \cdot \ell(e) > 1$ in the LP.

We have implemented the embedding algorithm and measured its performance on a variety of inputs. In the first experiment, we placed n nodes in a 15×15 square with a uniform distribution, and embed the largest connected component. The results are shown in the top part of Fig. 16, where each result is averaged over 50 experiments. In Fig. 16, *distance violation* is the number of non-adjacent node pairs that mistakenly have distance less than or equal to 1 in the embedding. d_{error} is the *minimum distance* between two non-adjacent embedded nodes that mistakenly have distance less than or equal to 1 in the embedding. (So $d_{error} \leq 1$ if such a pair of nodes exist; if no such node pair exists, we let $d_{error} = 1$). *Extra crossing* is the number of edge pairs that do not cross in the true UDG but mistakenly cross each other in the embedding. Note that the other criteria for embedding are guaranteed to be satisfied by the LP method: the *edge-length constraint* guarantees that every edge has length at most 1; the *cycle constraint* guarantees that all the angles between adjacent edges are as specified; the *crossing-edge constraint* guarantees that any two edges that cross in the true UDG also cross in the embedding. In Fig. 16 some additional properties are displayed as well, where *order of graph* is the number of nodes in the embedded UDG, and *node degree* is the average degree of nodes. A typical embedding result is shown in Fig. 17.

In a second experiment, we place nodes in an annulus with external radius 7.5 and internal radius 2.5. The results are shown in the bottom part of Fig. 16. A typical embedding result is shown in Fig. 18.

We can clearly see that the results are very good. Compared to previous results on embedding in the literature, our results can be seen to have superb performance without using landmarks [2] or edge-length information [13], even when the edges in the unit disk graphs are sparse. The number of non-adjacent node pairs having distance less than or equal to 1 in the embedding is very small, and even for such node pairs, their distances are close to 1. The number of incorrect edge crossings in the embedded graphs is very close to 0. We have also conducted experiments with many other inputs and in areas of other shapes, and the results have been consistently very good. Therefore the LP-based method does produce an almost truthful localization for sensor networks.

	network in square				
	order of graph	node degree	distance violation	d_{error}	extra crossing
$n = 200$	33.22	3.6422	0.80	0.9728	0.00
$n = 400$	337.96	5.4512	9.68	0.7642	0.50
$n = 600$	596.82	7.9110	6.50	0.8714	0.68
$n = 800$	799.64	10.5237	1.60	0.9568	0.10
$n = 1000$	999.94	13.1944	0.68	0.9601	0.00
	network in annulus				
	order of graph	node degree	distance violation	d_{error}	extra crossing
$n = 200$	59.76	4.1810	1.70	0.9368	0.00
$n = 400$	397.30	7.4084	6.62	0.8426	0.42
$n = 600$	599.88	11.0106	0.90	0.9570	0.08
$n = 800$	799.88	14.6423	0.10	0.9909	0.00
$n = 1000$	1000.00	18.2822	0.00	1.0000	0.00

Figure 16. Performance of embedding unit disk graphs deployed in a square and an annulus. Each result is averaged over 50 experiments.

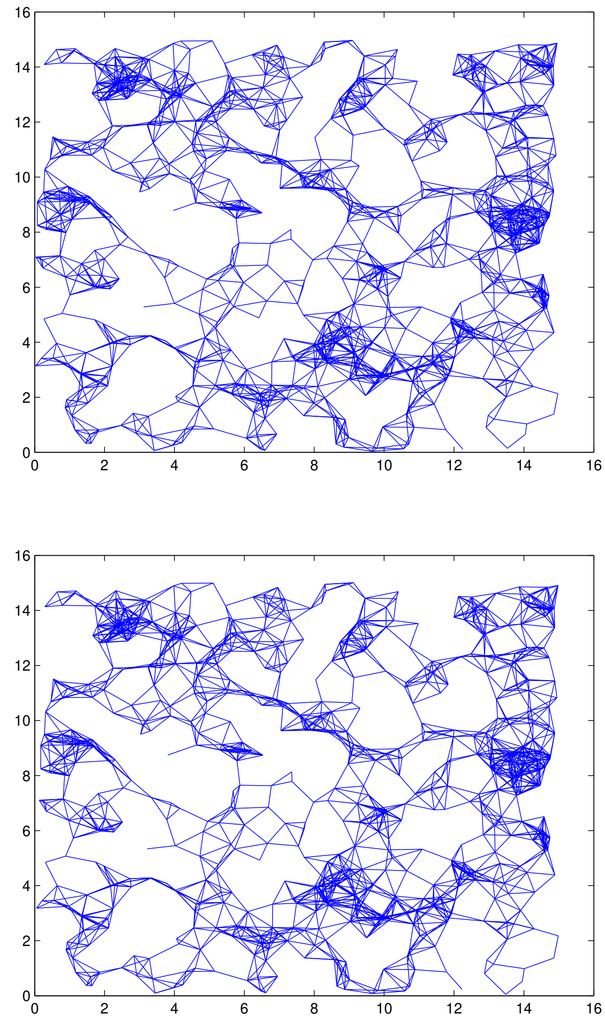

Figure 17. The unit disk graph of 597 nodes randomly deployed inside a 15×15 square. Top: the original UDG. Bottom: embedding by LP.

6.2 Geographical routing and approximate shortest path routing

In this section we examine the performance of routing schemes on the embedding of a unit disk graph by the linear program. In particular, given a unit disk graph with angle constraints, we find an embedding by the LP. Further, we embed the planar spanner constructed in the previous section using only local angle information. In particular we exclude the edges not in the spanner from the embedded UDG; if two edges still cross, we arbitrarily exclude one (this second step is heuristic). We run a particular geographical routing protocol (GPSR) and the approximate shortest path routing on this embedded UDG and its planar subgraph and compare the performance with that on the original (true) embedding.

Geographical routing and the approximate shortest path routing have their special requirements that differ from the criteria commonly used for localization. Geographical routing constantly makes local decisions on choosing the next hop, so it is important that the ranking of the distances from nearby nodes to any faraway destination is well maintained by the embedding. The graph-labelling-based approximate shortest path routing routes along shortest paths in planar spanners, so the distances between all pairs of nodes, adjacent or not, need to be well maintained in the embedding. Those requirements are global structures of a localization and differ from the comparatively more local criteria commonly used for localization — whether the node distance passes the threshold of 1, or whether two edges incorrectly cross or not cross. The success of the two routing algorithms in the embedded graphs shows the power of local angle information for routing, which reaches beyond the common objectives of network localization.

We experiment on sensor networks embedded with the LP approach, and compare its routing performance to that of the sensor networks with true coordinates. In the first experiment, we place n nodes in a 15×15 square with a uniform distribution, and embed the largest connected component. Then 20 source-destination node pairs are randomly selected, and routing is performed for each pair. We measure the Euclidean length (resp., number of hops) of a routing path, as well as that of the routing path with the same source-destination pair in the graph with true coordinates; we call the ratio between them the *length distortion* (resp., *hop distortion*), and denote it by D_l (resp., D_h). (Note that the Euclidean length of a routing path performed on the embedded graph should still be measured based on the true Euclidean lengths of its edges.) In the second experiment nodes are placed in an annulus with external radius 7.5 and internal radius 2.5, while other conditions are unchanged. The results for GPSR and approximate shortest path routing (ASPR) are shown in Fig. 19, where each result is averaged over 50 experiments and 20 source-destination pairs in each experiment.

Fig. 19 shows that for GPSR and ASPR, they both have the same routing performance in the embedded networks as in the true networks, both in terms of length and hops. In fact, a detailed study showed us that most of the time, the routing routes in the embedded networks are identical to their counterparts in the true networks. We have also conducted experiment for many other inputs and in areas of other shapes, and the results have been consistently as good. Thus not only does the LP give very good local embedding, i.e., neighboring nodes are close and non-neighboring nodes are far away, but it also gives a quite accurate global view such that geographical routing and approximate shortest path routing on the embedded graph are almost identical to those on the original (true) embedding.

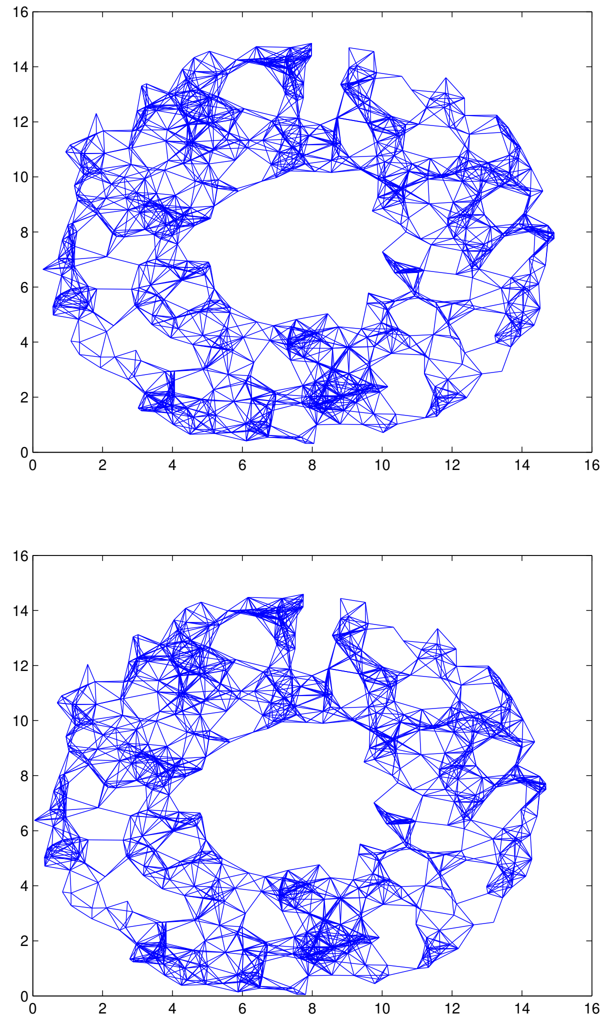

Figure 18. The unit disk graph of 600 nodes randomly deployed inside an annulus. Top: the original UDG. Bottom: embedding by LP.

6.3 Variations

In this subsection, we address the localization problem with noisy angle measurements and with sensor networks modelled as quasi-unit disk graphs. The simulation we have shown so far assumes that the angles are measured accurately. In practice measurement errors are inevitable. The modelling of sensor networks as UDG can be inaccurate, too, because network links can be lost due to noise, signal interference or obstacles, and the transmission ranges of directional antennas are not circles. A more realistic model for sensor networks is called quasi-unit disk graphs, where a pair of nodes have an edge for sure if their distance is no more than $\alpha \leq 1$, don't have an edge if their distance is more than 1 apart, and may or may not have an edge if their distance is between α and 1 [18].

We will show by simulation that the embedding algorithm by LP is robust to measurement errors and network models. Noisy measurements will introduce inconsistency in the input data. For example, the measured inner angles of a cycle may not sum up to the correct value. Thus we modify the constraints of LP accordingly.

	network in square				
	$n =$ 200	$n =$ 400	$n =$ 600	$n =$ 800	$n =$ 1000
GPSR D_l	1.1549	1.0011	1.0000	1.0000	1.0000
GPSR D_h	1.1403	1.0007	1.0000	1.0000	1.0000
ASPR D_l	1.0000	1.0000	1.0001	1.0000	1.0000
ASPR D_h	1.0000	1.0000	1.0000	1.0000	1.0000
	network in annulus				
	$n =$ 200	$n =$ 400	$n =$ 600	$n =$ 800	$n =$ 1000
GPSR D_l	1.0580	1.0078	1.0014	1.0000	1.0000
GPSR D_h	1.0575	1.0099	1.0012	1.0000	1.0000
ASPR D_l	1.0000	1.0001	1.0000	1.0000	1.0000
ASPR D_h	1.0000	0.9989	1.0000	1.0000	1.0000

Figure 19. Length distortion and hop distortion for GPSR and ASPR, averaged over 50 experiments and 20 source-destination pairs per experiment.

Specifically, the *cycle constraint* is modified to be

$$|\sum_{i=1}^{p} \ell(e_i) \cos\theta_{e_i}| \leq \varepsilon \cdot \sum_{i=1}^{p} |\sin\theta_{e_i}|,$$

and

$$|\sum_{i=1}^{p} \ell(e_i) \sin\theta_{e_i}| \leq \varepsilon \cdot \sum_{i=1}^{p} |\cos\theta_{e_i}|,$$

where ε is an additional variable. The *non-adjacent node pair constraint* is modified to be $\ell(e_1) + \ell(e_2) > \alpha$ due to the Quasi-UDG property. The *edge-length constraint* is maintained, and the *crossing-edge constraint* is discarded. The objective function is modified to be minimizing $\varepsilon - \min_e \ell(e)$. A solution of the LP gives the edge lengths; then we randomly choose a spanning tree of the network, and use its edge lengths and measured angles to determine node positions. The random spanning tree is generated a few times, and the one that gives comparatively better embedding performance is picked. In the final step, minor local adjustments in the node positions and the network size scaling factor are used to further improve the embedding result.

In the following experiment, we assume that each node measures the direction of an incident edge with an error uniformly distributed in $[-\Delta, +\Delta]$. As a result, the error of a local angle between adjacent edges can be as large as 2Δ or -2Δ. For the quasi-UDG model, we assume that for two nodes whose distance d is between α and 1, there is an edge with probability $\frac{1-d}{1-\alpha}$. Such a model has the property that nearby nodes are more likely to have edges. We place 225 nodes in a 10×10 square. For the node positions we use the *grid with perturbation* model. Specifically, The position of a node indexed by (i, j) is $(i \cdot \delta + \gamma \cos\varphi, j \cdot \delta + \gamma \sin\varphi)$, where δ is the grid's step size, γ is an i.i.d. Gaussian variable with mean 0 and variance σ_p^2, and φ is an i.i.d. variable uniformly distributed in the range $[0, 2\pi]$. For this experiment, $\delta = \frac{2}{3}$ and $\sigma_p = 1.5$. The performance measurements include the *total distance violation*, which is the number of node pairs that are adjacent but have embedding distance more than 1 or that are non-adjacent but have embedding distance less than α, and the *total crossing violation*, which is the number of edge pairs that do not actually cross but mistakenly cross in the embedding or the other way around. The results are shown in Figure 20. Each result is averaged over 50 experiments. A typical result is shown in Figure 21.

Figure 20 shows that the embedding algorithm by LP is quite robust to noisy measurements of angles and the quasi-UDG model. The values of total distance violation and crossing violation are

	total distance violation				
	$\Delta = 1°$	$\Delta = 2°$	$\Delta = 3°$	$\Delta = 4°$	$\Delta = 5°$
$\alpha = 0.8$	29.66	34.52	42.32	51.90	54.50
$\alpha = 0.6$	17.36	17.56	20.08	24.46	27.76
$\alpha = 0.4$	7.86	9.08	9.18	9.44	10.18
$\alpha = 0.2$	4.00	3.08	4.08	4.30	5.22

	total crossing violation				
	$\Delta = 1°$	$\Delta = 2°$	$\Delta = 3°$	$\Delta = 4°$	$\Delta = 5°$
$\alpha = 0.8$	69.46	77.58	90.98	128.26	134.24
$\alpha = 0.6$	35.72	38.68	38.96	52.60	57.80
$\alpha = 0.4$	11.94	13.68	16.10	16.38	16.96
$\alpha = 0.2$	6.98	5.60	7.98	9.26	9.56

Figure 20. Performance of embedding quasi-UDG with noisy angle measurements. Each result is averaged over 50 experiments.

substantially greater than those in Figure 16, but they are not large considering the size of the graph. And a detailed study shows that the global structure of the graphs is maintained quite well, even though in our experiments no landmarks are used to help fix the large-scale structure. That can also be seen from Figure 21. We have conducted experiments with many other inputs and different quasi-UDG models and measurement error models, and the results have been very consistent.

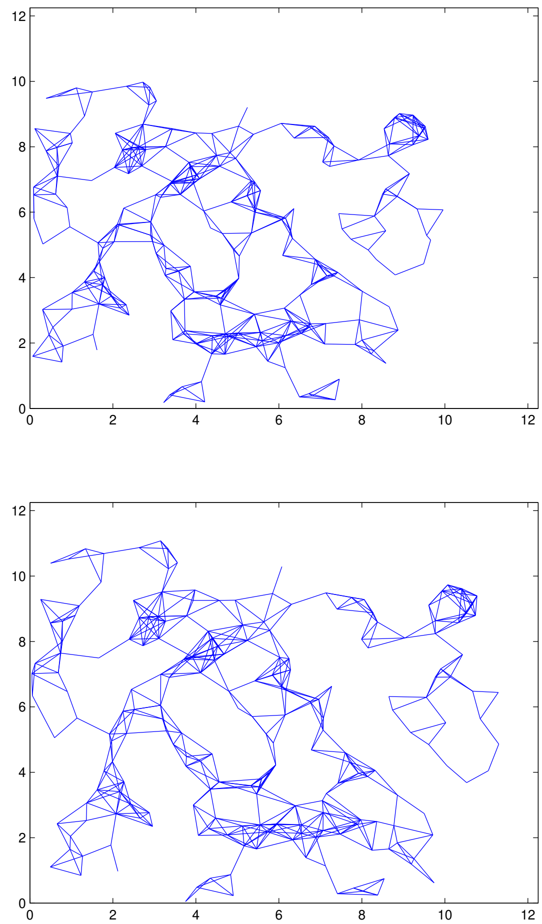

Figure 21. Embedding a quasi-UDG, with $\alpha = 0.8$ and $\Delta = 3°$. Top: the original quasi-UDG. Bottom: embedding by LP. The total distance violation is 40; the total crossing violation is 89.

7 Summary and future work

In this paper we studied embedding a unit disk graph in the plane with angle constraints. We show theoretically that this problem is actually NP-hard. We also propose a solution based on linear programming that gives very good results in practice. This work raises a few open questions. For example, it's unknown whether one can find an algorithm that gives a good approximate embedding with theoretical bounds in the worst case. Also, the linear program mentioned in this paper is a centralized algorithm, but in practice distributed localization methods are more desirable.

Acknowledgements: This work was supported in part by the Lee Center for Advanced Networking at the California Institute of Technology, and by NSF grant CCR-TC-0209042.

References

[1] J. Aspnes, D. Goldenberg, and Y. R. Yang. On the computational complexity of sensor network localization. In *The 1st International Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS)*, pages 32–44, 2004.

[2] P. Biswas and Y. Ye. Semidefinite programming for ad hoc wireless sensor network localization. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, pages 46–54, 2004.

[3] I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer-Verlag, 1997.

[4] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. In *DialM '99: Proceedings of the 3rd International Workshop on Discrete Algorithms and Mehtods for Mobile Computing and Communications*, pages 48–55, 1999.

[5] H. Breu and D. G. Kirkpatrick. Unit disk graph recognition is NP-hard. *Computational Geometry. Theory and Applications*, 9(1-2):3–24, 1998.

[6] V. W. Bryant. Straight line representation of planar graphs. *Elementary Mathematics*, 44:64–66, 1989.

[7] M. Bădoiu, E. D. Demaine, M. T. Hajiaghayi, and P. Indyk. Low-dimensional embedding with extra information. In *Proceedings of the 20th Annual Symposium on Computational Geometry*, pages 320–329, 2004.

[8] L. Doherty, L. E. Ghaoui, and S. J. Pister. Convex position estimation in wireless sensor networks. In *IEEE Infocom*, volume 3, pages 1655–1663, April 2001.

[9] I. Fáry. On straight line representations of planar graphs. *Acta Scientiarum Mathematicarum (Szeged)*, 11:229–233, 1948.

[10] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. Complex behavior at scale: An experimental study of low-power wireless sensor networks. Technical Report UCLA/CSD-TR 02-0013, UCLA, 2002.

[11] J. Gao, L. J. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Geometric spanner for routing in mobile networks. In *MobiHoc '01: Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 45–55, 2001.

[12] C. Gavoille, D. Peleg, S. Pérennes, and R. Raz. Distance labeling in graphs. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 210–219, 2001.

[13] C. Gotsman and Y. Koren. Distributed graph layout for sensor networks. In *Proceedings of the International Symposium on Grpah Drawing*.

[14] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning Systems: Theory and Practice*. Springer, 5 edition, 2001.

[15] B. Karp and H. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *MobiCom '00: Proceedings of the 6th ACM Annual International Conference on Mobile Computing and Networking*, pages 243–254, 2000.

[16] F. Kuhn, T. Moscibroda, and R. Wattenhofer. Unit disk graph approximation. In *Proceedings of the 2004 Joint Workshop on Foundations of Mobile Computing*, pages 17–23, 2004.

[17] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: of theory and practice. In *PODC '03: Proceedings of the 22nd Annual Symposium on Principles of Distributed Computing*, pages 63–72, New York, NY, USA, 2003. ACM Press.

[18] F. Kuhn and A. Zollinger. Ad-hoc networks beyond unit disk graphs. In *Proceedings of the 2003 Joint Workshop on Foundations of Mobile Computing*, pages 69–78, 2003.

[19] X.-Y. Li, G. Calinescu, and P.-J. Wan. Distributed construction of planar spanner and routing for ad hoc networks. In *IEEE INFOCOM*, pages 1268 – 1277, 2002.

[20] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust distributed network localization with noisy range measurements. In *Proceedings of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*.

[21] T. Moscibroda, R. O'Dell, M. Wattenhofer, and R. Wattenhofer. Virtual coordinates for ad hoc and sensor networks. In *Proceedings of the 2004 Joint Workshop on Foundations of Mobile Computing*, pages 8–16, 2004.

[22] D. Niculescu and B. Nath. Ad hoc positioning system (APS). In *IEEE GLOBECOM*, pages 2926–2931, 2001.

[23] D. Niculescu and B. Nath. *Ad hoc* positioning system (APS) using AOA. In *IEEE INFOCOM*, volume 22, 2003.

[24] D. Niculescu and B. Nath. Error characteristics of ad hoc positioning systems (APS). In *MobiHoc '04: Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 20–30, 2004.

[25] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *MobiCom '00: Proceedings of the 6th ACM Annual International Conference on Mobile Computing and Networking*, pages 32–43, 2000.

[26] R. Rajaraman. Topology control and routing in ad hoc networks: a survey. *SIGACT News*, 33(2):60–73, 2002.

[27] A. Rao, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *MobiCom '03: Proceedings of the 9th ACM Annual International Conference on Mobile Computing and Networking*, pages 96–108, 2003.

[28] A. Savvides, C.-C. Han, and M. B. Strivastava. Dynamic fine-grained localization in *ad-hoc* networks of sensors. In *MobiCom '01: Proceedings of the 7th ACM Annual International Conference on Mobile Computing and Networking*, pages 166–179, 2001.

[29] A. Savvides and M. B. Strivastava. Distributed fine-grained localization in *ad-hoc* networks. submitted to *IEEE Transactions on Mobile Computing*.

[30] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz. Localization from mere connectivity. In *MobiHoc '03: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 201–212, 2003.

[31] A. M.-C. So and Y. Ye. Theory of semidefinite programming for sensor network localization. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2005.

8 Appendix

Now we prove that a $\sqrt{2}$-approximate embedding is topologically equivalent with a valid embedding.

If there are four nodes A, B, C, D such that in the unit-disk graph G there are edges AB, CD, AD, we show that it's impossible to have AB, CD cross in a $\sqrt{2}$-approximate embedding $\mathcal{E}$, but not cross in a valid embedding $\mathcal{E}^*$.

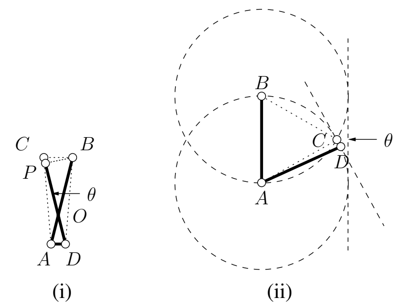

Figure 22. (i) A $\sqrt{2}$-approximate embedding $\mathcal{E}$; (ii) A valid embedding $\mathcal{E}^*$.

Without loss of generality we assume that edge AB is no shorter than CD and the embedding $\mathcal{E}$ looks like Figure 22 (i). First, if AB, CD cross in $\mathcal{E}$, then $\angle BAD + \angle CDA < \pi$. Otherwise AB, CD will never cross in any embedding preserving the angles. Notice that the angle θ between line AB, CD doesn't change for any embedding preserving the angles. We argue that θ is at most $\pi/6$ if we can find a valid embedding $\mathcal{E}^*$ such that AB, CD don't cross. See Figure 22 (ii). Specifically, in a valid embedding $\mathcal{E}^*$ there are no edges AC in the unit disk graph. Thus $\mathcal{E}^*(C)$ is outside the unit disk centered at $\mathcal{E}^*(A)$. $\mathcal{E}^*(B), \mathcal{E}^*(D)$ are inside the unit disk centered at $\mathcal{E}^*(A)$. It's not hard to see that the angle θ achieves the maximum $\pi/6$ when $\mathcal{E}^*(A)$. $\mathcal{E}^*(B)$, $\mathcal{E}^*(D)$ are exactly of distance 1 pairwise apart and D is arbitrarily close to C such that CD is arbitrarily close to the tangent at C. So $\theta \leq \pi/6$.

In a $\sqrt{2}$-approximate embedding $\mathcal{E}$, suppose O is the intersection of edges AB, CD. $\angle BOC = \theta \leq \pi/6$. Since the length of BC, BD, CA are all greater than $\sqrt{2}/2$, the angles $\angle ACB$, $\angle CBD$ are both less than $\pi/2$. Thus the angle $\angle BCD$, $\angle CBA$ are less than $\pi/2$ as well. Assume without loss of generality that BO is longer than CO. We take the perpendicular line through B to the line CO and denote the intersection as P. P must be on the interior of line segment CO since $\angle OCB < \pi/2$. Thus the length of BC achieves the maximum when CO has the same length of BO. Thus $d(\mathcal{E}(B), \mathcal{E}(C)) \leq 2d(\mathcal{E}(B), \mathcal{E}(O)) \sin(\pi/12) \leq 2\sin(\pi/12) \approx 0.52$. This contradicts with the assumption that BC has length at least $\sqrt{2}/2$.