Sensor Node Localization Using Uncontrolled Events

Ziguo Zhong, Dan Wang and Tian He
Department of Computer Science and Engineering, University of Minnesota
{zhong, tianhe}@cs.umn.edu, {wang0889}@umn.edu

Abstract

Many event-driven localization methods have been proposed as low cost, energy efficient solutions for wireless senor networks. In order to eliminate the requirement of accurately controlled events in existing approaches, we present a practical design using totally uncontrolled events for stationary sensor node positioning. The novel idea of this design is to estimate both the event generation parameters and the location of each sensor node by processing node sequences easily obtained from uncontrolled event distribution. To demonstrate the generality of our design, both straight-line scan and circular wave propagation events are addressed in this paper, and we evaluated our approach through theoretical analysis, extensive simulation and a physical testbed implementation with 41 MICAz motes. The evaluation results illustrate that with only randomly generated events, our solution can effectively localize sensor nodes with excellent flexibility while adding no extra cost at the resource constrained sensor node side. In addition, localization using uncontrolled events provides a nice potential option of achieving node positioning through natural ambient events.

1 Introduction

Wireless sensor networks (WSN) shows more and more popularity for both military and civil applications [6, 23, 14, 26, 28, 7, 10]. The geographical location information of each sensor node in the network is critical for many applications because users need to know not only what happened, but also where interested events happened. In addition, some routing protocols [12, 13] are built under the assumption that geographic parameters of sensor nodes are available. However, sensor node localization is still one of the challenging problems because of extremely demanding requirements for low cost, tiny size and high energy efficiency at the sensor node side.

Many excellent ideas have been proposed for addressing node positioning in sensor networks. Most of them can be categorized into three classes: (i) range-based localization [16, 2, 19, 5, 29, 21, 11, 32, 3, 8]; (ii) range-free localization [15, 4, 9, 18, 1, 22]; and (iii) event-driven localization [20, 24, 25, 17, 33].

Range-based localization approaches are built on top of distance or angle measurements among sensor nodes in the networks. Range-based methods either are costly for using per-node ranging hardware, or need careful in-field calibration and environment profiling [22, 31, 9]. Range-free sensor node localization doesn't need any forms of ranging. Instead, the location of each node is estimated based on knowledge of proximity to the anchor nodes whose location information is known [4, 9, 18]. Range-free localization methods normally have low accuracy, highly depending on the density and distribution of the anchor nodes.

Event-driven localization makes use of localization events which are generated and propagate across the area where sensor networks are deployed. With known time-spatial relationship embedded in the event distribution, the location of each sensor node can be obtained by mapping the time of event detection with the event position at that time instance. [24, 33, 20, 17] work in this fashion. Since sensor nodes only need to detect the events and report the detections, event-driven approaches apply an asymmetric system architecture [24] which significantly reduces the cost and energy consumption at the resource constrained sensor node side. Two generations of event-driven localization methods have been proposed over past several years. The first generation methods (e.g., Spotlight [17, 24, 20]) obtain accurate location results, requiring precisely-controlled events. The second generation methods (e.g., MSP [33]) relax the requirement of precisely-controlled events into partially-controlled events.

As an important step further, this work, for the first time, demonstrates the possibility to accomplish event-driven localization with totally uncontrolled events. Localization using uncontrolled events has two obvious benefits. First of all, simple event generation mechanisms can be applied to make the system very flexible and convenient to work with. Secondly, non-artificial natural events could possibly be utilized for localization purpose. With totally uncontrolled localization events, our solution estimates the location of each sensor node by processing node sequences easily obtained from event distributions. Specifically, our design has following features:

- To the best of our knowledge, this paper is the first one to do sensor node localization using totally uncontrolled events. Localization using uncontrolled events significantly improves the flexibility and convenience for system deployment.
- As an event-driven localization approach, no additional ranging hardware or in-field calibration is needed at the resource constrained sensor node side.

The rest of the paper is organized as follows. Section 2 briefly surveys previous localization methods. Section 3 presents preliminary system overview. Section 4 details the system design. In Section 5, we give examples and explanations for wave propagation events. System overhead is analyzed in Section 6. Section 7 illustrates simulation results and Section 8 reports an implementation of this system on our Mirage testbed. Finally, Section 9 concludes the whole paper.

2 Related Work

Range-based approaches [16, 2, 19, 5, 29, 21, 11, 32, 3, 8] are based on distance or angle measurements among sensor nodes in the network. Then the location of the each sensor node can be obtained from geometric computation and estimation. Many range-based methods tried to eliminate expensive or energy hungry ranging devices, e.g., ultrasound, infrared emitter/receiver, laser beam scanner, by assuming and using RSSI (Receive Signal Strength Indicator) [2, 32, 22, 29] with noise filtering. However, recent empirical study [30, 31, 29] concluded that unless careful calibration and environment profiling were accomplished, radio ranging irregularity [9, 22, 31] is detrimental to the system accuracy.

Range-free solutions [15, 4, 9, 18, 1, 22] try to estimate the location of each sensor node based on the information of geographic proximity to the anchor nodes whose positions are known. Centroid [4], APIT [9], APS [18] are typical examples for this category. In those solutions, sensor nodes estimate their locations according to the known position information of the surrounding anchor nodes. Different anchor combinations help to narrow the location area where a normal node possibly be located. Anchor-based approaches normally demand high and uniformly distributed anchor density in the network so as to achieve good positioning accuracy. In reality, it is highly appreciated to use as few anchor nodes as possible in order to reduce system cost.

Keeping system cost in mind, researchers developed several event-driven localization methods [20, 24, 25, 17, 33]. Lighthouse [20], SpotLight [24] and indoor experiments using laser events [17] provide us with examples for achieving good localization accuracy without any anchor. The basic idea of those event-driven methods is to make use of time-spatial relationships embedded in each event distribution. Given event detection time at each sensor node, centralized localization engine could map the detection time to the position of the corresponding node. Although asymmetric system architecture [24] shifts the resource cost from the sensor nodes to the event-generation device, which significantly brings down the overall system cost, we argue that in reality, accurate control of localization event distribution could be difficult, costly or time-consuming to achieve for large area outdoor scenarios.

MSP [33] brings in anchor nodes for event-driven localization. Without depending on rigid time-spatial relationship carried by the localization event distribution, MSP obtains possible location area of each sensor node by processing multiple one-dimensional node sequences within which relative position information along the event propagation direction is embedded. In MSP, only a few number of anchor nodes are necessary, and the requirement for accurate event distribution control is removed. However, the setup of MSP assumes that accurate control of the event generation is available, e.g., straight-line scan with certain angle. Therefore, MSP is actually a hybrid solution combining anchors and semi-controlled localization events.

In this paper, we proposed a sensor node localization system design based on totally uncontrolled events. The solution in this paper eliminates the control over event generation and event distribution. Therefore, it could significantly improve the flexibility and convenience for system deployment.

3 System Overview

Figure 1 shows a snapshot of an optical straight-line scan event generated by projectors of our *Mirage* testbed. There are two levels of control over the localization events: (i) control of how to generate the event, e.g., scan at certain angle in Figure 1. We call this type of control as control of *event generation parameter*; (ii) control over event propagation, e.g., keep constant scan line-speed on the testbed, which is called control of *event distribution*. For randomly generated events, neither of the above controls is possible, and in reality, both levels of control could be hard and costly to achieve.

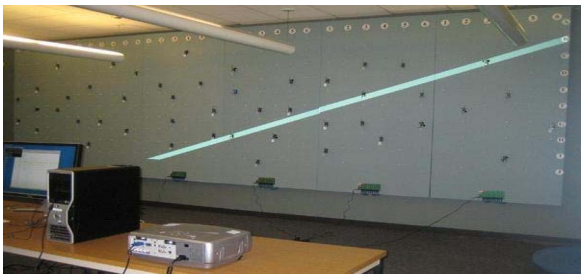
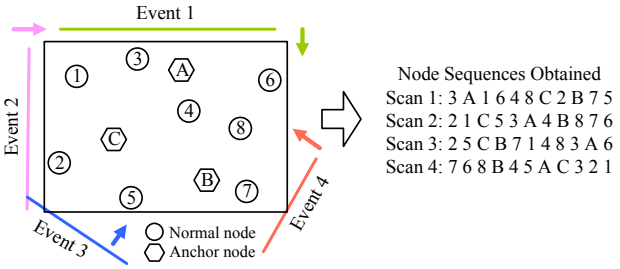
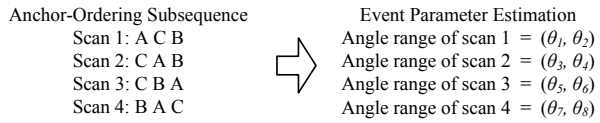


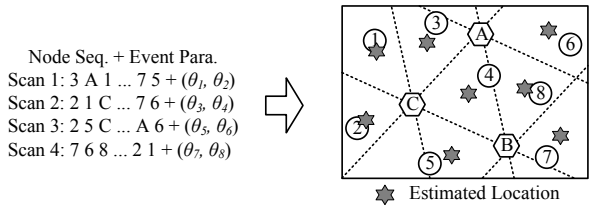
Figure 1. Straight-line Scan Using Mirage Testbed



(a) Node sequences obtained from uncontrolled events



(b) Event parameter estimation using anchors in the sequences



(c) Map splitting and location estimation

Figure 2. System Overview

The basic idea for localization using uncontrolled events is to estimate the event generation parameter using anchors in the field, and then shrink the possible location area of each normal node according to the estimated event parameter. In brief, as it is shown in Figure 2, the system works as follows. First, certain type of events are generated in the network, e.g., straight-line laser beam scan with uncontrolled angle, direction and speed. As each event propagates, sensor nodes detect the event sequentially at different time instances, which naturally gives out an ordering of in-the-field nodes called *node sequence*. For exam-

ple, as shown in Figure 2(a), a top-down scan event generates node sequence (3 A 1 6 4 8 C 2 B 7 5). Here we use uppercase letters (e.g., A, B, C) denote anchor nodes and numbers (e.g., 1, 2, 3) denote normal nodes. Second, node sequences processing algorithms try to estimate the event generation parameter, e.g., possible scan angle range, by processing ordered anchor subsequences which can be extracted directly from node sequences as shown in Figure 2(b). Third, processing a node sequence with its corresponding estimated event generation parameter, the whole map can be divided into lots of small parts. Each normal sensor node obtains a possible location area, which is composed of different single or multiple parts, according to their ranks in the node sequence. With multiple events, final location area of a normal node could be shrunk dramatically by extracting the joint region of the possible location areas given by all the node sequences. Thus the estimated position could be got from a relatively small location area to achieve good localization accuracy (Figure 2(c)).

The design we propose in this paper can be extended from two-dimensional(2D) map localization to three-dimensional (3D) case. Due to the space constrain of the paper, our discussion focuses on 2D localization examples. In the following sections, for the sake of clarity, straight-line scan is used as an example localization event. We will explain later about how other types of events (e.g., circular wave propagation) can also be used similarly.

4 System Design

This section explains the algorithms used for localization using uncontrolled events. There are two steps: (i) event generation parameter estimation (Section 4.1), and (ii) location area estimation (Section 4.2). Every sensor node's sequential detection of the event generates an ordering of all the in-field nodes, namely node sequence. Event generation parameter estimation is achieved by processing the anchor subsequences (ordering of anchor nodes) of the node sequences. Then location areas of normal nodes are estimated by processing the node sequences along with the estimated event generation parameters. At the end of this section, intergraded algorithm is presented (Section 4.3).

4.1 Event Generation Parameter Estimation

This subsection introduces event generation parameter estimation for uncontrolled localization events. The idea is to make use of the ordering of anchors embedded in the node sequence, so as to extract the possible range of the event generation parameter.

For a straight-line scan event, the event generation parameter to be estimated is the scanning angle. Given three anchors not in a line, the ordering of these three anchors reflects both the scan direction and the possible range of the scan angle. As it is shown in Figure 3, if a straight-line scans from top to bottom, a node sequence: (3 A 1 6 4 8 C 2 B 7 5) is obtained. Reversely, given this node sequence, anchor ordering subsequence (A C B) can be extracted. Known the location of anchor A, B, C, by careful observation, we can easily conclude that (i) the scan must be conducted from top to bottom; (ii) the scan angle must be within the range of θ ($\angle ACB$) indicated in Figure 3, namely the possible scan angle of of this straight-line scan must be within the range of (θ_1, θ_2). Any scan angle beyond θ can

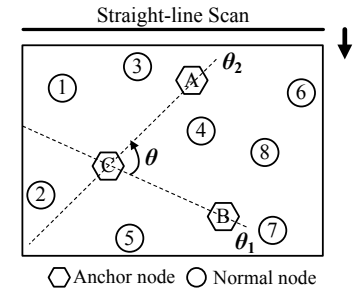


Figure 3. Estimate Angle Range by Intuition

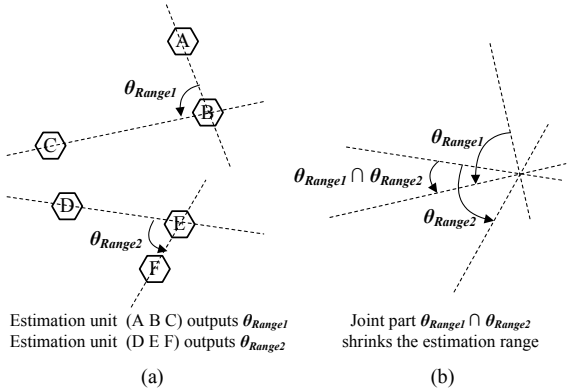


Figure 4. Extract the Joint Part of Estimations

not output an anchor ordering of (A C B).

For a straight-line scan event, as it is shown in the above example, a three-anchor ordering can give out an event generation parameter estimation, namely a possible angle range for the scan. We call such a three-anchor ordering as an *estimation unit* for the straight-line scan event. For multiple anchors existing in a node sequence, each combination of three anchors creates an estimation unit, which is able to output an estimation. Final result for event parameter estimation is obtained by extracting the joint part of results given by all the estimation units. This is because all of them need to be satisfied.

An example is shown in Figure 4, an anchor subsequence (A B C $\cdots$ D E F $\cdots$) containing multiple anchors is obtained from a node sequence. Estimation unit (A B C) gives a scan angle estimation θ_{Range1}, and another estimation unit (D E F) gives another estimation θ_{Range2}, illustrated in Figure 4(a). Then the joint part $\theta_{Range1} \cap \theta_{Range2}$ is an estimation with smaller interval (uncertain range), as it is shown in Figure 4(b). With increasing number of anchor nodes, the uncertain range of the event generation parameter estimation shrinks dramatically, and accurate estimation can be achieved.

With increasing number of anchors, computation complicity also increases. A quick conclusion is that for a node sequence with n anchors, there are at most $C_n^3 = \frac{n^3-3n^2+2n}{6} = O(n^3)$ different three-anchor estimation units. Actually, only $O(n)$ estimation units are helpful for event generation parameter estimation. Figure 5 shows a simple example for the reason. We can see from the figure that if A and B are the first two anchors in a three-anchor estimation unit, only choosing anchor C, which is

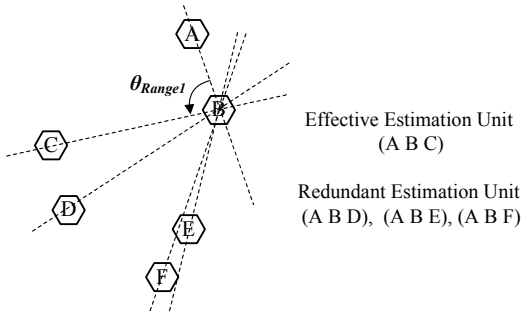


Figure 5. Example of Redundant Estimation Units

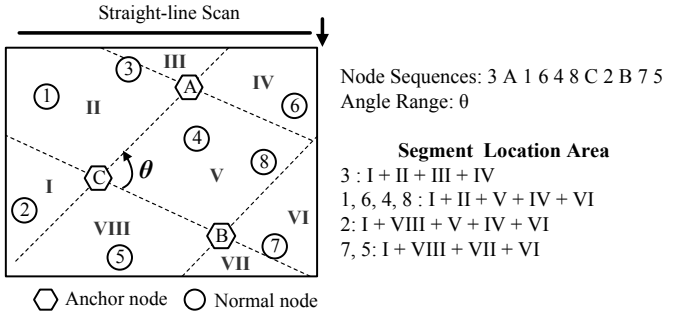


Figure 6. Example of Area Dividing

next anchor of anchor B in the anchor subsequence, generates an *effective estimation unit*. This is because estimation units composed of A, B and any other anchors beyond C in the anchor subsequence would create wider angle-range estimations covering θ_{Range1} which is given by estimation unit (A B C). As shown in the figure, $\angle ABD$, $\angle ABE$, and $\angle ABF$, given by estimation unit (A B D), (A B E) and (A B F) respectively, are actually *redundant* due to their wider range compared to $\theta_{Range1}(\angle ABC)$. Thus, only three consecutive anchor nodes form an effective estimation unit for the straight-line scan event.

Generally, if l anchors are needed in one estimation unit, with n anchor nodes in the node sequence, the number of effective estimation unit is at most $n-l+1$, namely only $O(n)$ estimation units are necessary for computation.

Algorithm 1 depicts the computation architecture for event generation parameter estimation. For each node sequence, all effective estimation units are used to estimate the event generation parameter independently, then final parameter estimation $\widehat{\theta}$ is obtained by extracting the joint part of all estimations. Specifically, line 1 initializes $\widehat{\theta}$ to be the scope of all possible values of the event generation parameter. For example, $(-\frac{\pi}{2}, +\frac{\pi}{2})$ for straight-line scan angle. All effective estimation units are obtained and processed between line 2 and 6. Line 3 obtains an unprocessed effective estimation unit $Unit_i$. Line 4 computes corresponding estimation $\widehat{\theta}_i$. Then, $\widehat{\theta}$ is shrunk by extracting the joint part of previous $\widehat{\theta}$ and the newly obtained $\widehat{\theta}_i$ at line 5.

Algorithm 1 Event Generation Parameter Estimation

Input: A node sequence: *NodeSeq*

Output: Estimated event generation parameter: $\widehat{\theta}$

1: $\widehat{\theta}$ = All possible values of the parameter;
2: **repeat**
3: $Unit_i$ = GetUnusedEstimationUnit(*NodeSeq*);
4: $\widehat{\theta}_i$ = EventParameterEstimation($Unit_i$);
5: $\widehat{\theta} = \widehat{\theta} \cap \widehat{\theta}_i$;
6: **until** all effective estimation units in *NodeSeq* are used;

4.2 Location Area Estimation

With estimated event generation parameter range $\widehat{\theta}$ given by Algorithm 1, cooperating with the original node sequence, the whole area can be divided into multiple parts. For each normal node, its rank in the node sequence determines a possible area in which it is located.

Figure 6 shows the basic idea of area dividing according to a node sequence obtained from a straight-line scan event. As it is shown in the figure, a node sequence (3 A 1 6 4 8 C 2 B 7 5) is obtained from a top-down event. Angle range $\widehat{\theta} = \theta$ can be estimated from anchor subsequence (A C B). According to the location of anchor A, B and C, and the estimated angle range θ, the whole area is divided into 8 parts (I, II, $\cdots$, VIII). In the node sequence, node 3 is ahead of anchor A. By carefully observation, we can see that node 3 can only be located in the area which is the union of I, II, III and IV. This is because if node 3 is located within this area, it is possible to satisfy the obtained node sequence with a scan event whose scan angle is within the range of θ.

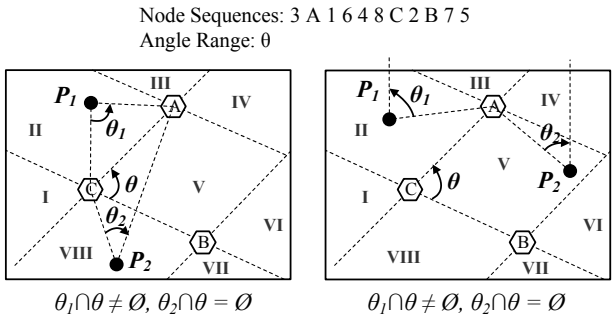
For a system containing n anchor nodes, according to the pie-cutting theorem [27], the whole area would be cut into $O(4n^2)$ small parts by a straight-line scan event, e.g., I, II, III, $\cdots$, in Figure 6. On the other hand, a node sequence could be divided into at most $n+1$ segments by n anchors. For example, in Figure 6, three anchors A, B, C divide the node sequence (3 A 1 6 4 8 C 2 B 7 5) into four segments: (3), (1 6 4 8), (2), and (7, 5). Normal nodes within the same segment share the same possible location area in terms of this cut. The possible area for each segment is a combination of multiple contiguous small parts. If multiple node sequences obtained from different events are processed, each normal node will get multiple possible location areas composed of diverse parts. Then, the joint area of all its possible areas is the final location area of this normal node. With increasing number of anchor nodes and increasing number of events, the location area for each normal node could be shrunk dramatically. Thus, effective sensor node localization can be achieved.

4.2.1 Estimation based on Grid-Based Sampling

Based on the intuitive analysis above, this subsection details the computation algorithm for location area estimation.

Modelling the whole area using a grid map, a pixel in the map stands for a differentiable position point. In order to find the location area for a segment s in the node sequence, all differentiable position points in the map are investigated. For each position point, we calculate the event generation parameter $\widetilde{\theta}$ satisfying the requirement that if a normal node locates at this position, it would appear within segment s. Then the required event generation parameter $\widetilde{\theta}$ and obtained event generation parameter estimation $\widehat{\theta}$ are compared. If they have overlapping portion, which means it is possible that the event satisfies $\widetilde{\theta}$, then this point is a possible position for segment s. Otherwise, this point is excluded from the location are of segment s.

Figure 7 illustrates two examples. In fact, there are two cases: (i) a segment is between two anchors in the node se-



(a) Segment between two anchors (b) First segment example

Figure 7. Example of Location Area Finding

quence, e.g., segment (1 6 4 8) in the node sequence; (ii) Either the first segment or the last segment, which has only one neighboring anchor in the node sequence, e.g., segment (3). Figure 7 (a) shows an example for segment (1 6 4 8) which follows case (i). Two position points P_1 and P_2 are investigated. As it is shown in the figure, $\widehat{\theta} = \theta$. For point P_1's being ranked between anchor A and C in a node sequence, the required scan angle range $\widetilde{\theta} = \theta_1$. Since $\theta_1 \cap \theta \neq \varnothing$, namely $\widetilde{\theta} \cap \widehat{\theta} \neq \varnothing$, P_1 is a possible position for the segment between anchor A and C. While for point P_2, the required scan angle range is $\widetilde{\theta} = \theta_2$. $\theta_2 \cap \theta = \varnothing$, namely $\widetilde{\theta} \cap \widehat{\theta} = \varnothing$, therefore P_2 is not in the location area for the segment between anchor A and C.

Figure 7 (b) shows an example for case (ii). P_1 and P_2 are investigated for segment (3) which has only one neighboring anchor A. Here it is a little bit tricky because one anchor is not sufficient to give out an angle range. Imaging there is another dummy anchor located far far away above anchor A, then this dummy anchor can be regarded as the other neighboring anchor for segment (3). Thus a vertical line can be used for the other boundary of $\widetilde{\theta}$. Similarly, because $\theta_1 \cap \theta \neq \varnothing$ and $\theta_2 \cap \theta = \varnothing$, we can conclude that P_1 is valid for segment (3) while P_2 is not.

Algorithm 2 illustrates how to compute possible location area for a segment in a node sequence. First of all, the possible area for a *Segment* is initialized to be $\varnothing$ (line 1 in Algorithm 2). Then, each position point in the *Map* is investigated to check whether it is possible for the segment (line 2 to 10). Specifically, line 4 and 5 get the predecessor anchor and successor anchor for this segment in the node sequence (one of them might be a dummy anchor). Line 6 computes the required event generation parameter range $\widetilde{\theta}$. If the real event generation parameter is within the range of $\widetilde{\theta}$, a normal node at this position point P would be ranked between the predecessor anchor and the successor anchor in the node sequence. Line 7 to 9 say that if $\widetilde{\theta}$ has common part with estimated event generation parameter $\widehat{\theta}$, then P is a possible location for this segment and be added to *Area*.

Algorithm 2 Location Area Estimation

Input: A segment in a node sequence *Segment*
Estimated event parameter $\widehat{\theta}$
Output: Estimated location area for this segment *Area*

```
1:  Area = ∅;
2:  repeat
3:     P = GetUnprocessedPositioinPoint( Map );
4:     Anchor1 = PreAnchor( Segment );
5:     Anchor2 = SucAnchor( Segment );
6:     θ̃ = RequiredRange( Anchor1, P, Anchor2 );
7:     if θ̃ ∩ θ̂ ≠ ∅ then
8:        Area = Area ∪ P;
9:     end if
10: until all position points in the map are processed;
```

4.3 Localization Algorithm

Combining Algorithm 1 for event generation parameter estimation, and Algorithm 2 for segment location area estimation, this section provides the overall system design depicted by Algorithm 3. In fact, this is a general computation architecture for uncontrolled event based sensor node localization.

The input of the system is the node sequences obtained from uncontrolled localization events, and the output is estimated location of normal nodes. Line 1 to line 12 process each node sequence one by one to shrink the location area of each normal node. Specifically, line 2 gets a unprocessed node sequence, and then line 3 estimates the event generation parameter from this node sequence using Algorithm 1. Line 4 to 11 process each segment in the node sequence one by one with two steps. First, line 6 estimates the location area for this segment using Algorithm 2, and then from line 7 to 10, all normal nodes within this segment refresh their location areas by extracting the joint part of their previous areas and currently estimated area. Finally, central of gravity is used for estimating the final location of each normal node from line 13 to 16.

Algorithm 3 Localization with Uncontrolled Events

Input: Multiple node sequences: *NodeSeqs*.
Output: Estimated location of normal nodes.

```
1:  repeat
2:     NodeSeq = GetUnprocessedSequence( NodeSeqs );
3:     θ̂ = Algorithm 1( NodeSeq );
4:     repeat
5:        Segment = GetUnproessedSegment( NodeSeq );
6:        Area = Algorithm 2( Segment, θ̂ );
7:        repeat
8:           Node = GetUnprocessedNode( Segment );
9:           LocationAreaShrink( Node, Area );
10:       until all normal nodes in the Segment are processed
11:    until all segments in NodeSeq are processed
12: until all node sequences in NodeSeqs are processed;
13: repeat
14:    Node = GetUnprocessedNode( NodeSeq );
15:    CentroidEstimation( Node );
16: until all normal nodes in NodeSeq are estimated;
```

5 Wave Propagation Event Example

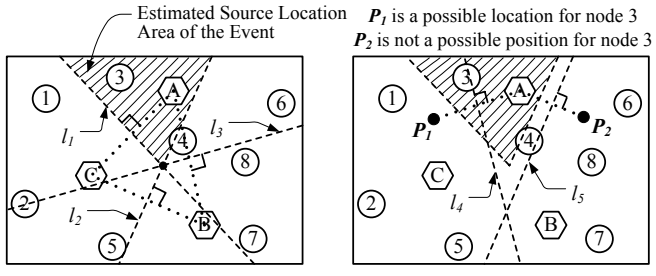
So far, the description of sensor node localization using uncontrolled events has been solely in the context of straight-line scan. In fact, algorithms proposed in this paper are conceptually independent of what types of events are utilized as long as node sequences can be obtained. Clearly, we can also support wave-propagation-based events (e.g., ultrasound propagation, air blast

propagation), which are polar coordinate equivalences of the straight-line scans in the Cartesian coordinate system. This section gives a brief explanation of wave propagation-based situation. Without losing generality, we have made the following assumptions:

- The wave propagates uniformly in all directions, therefore the propagation has a circle frontier surface. Node sequence processing does not rely on an accurate time-spatial relationship, a certain distortion in wave propagation is tolerable. If any directional wave is used, the propagation frontier surface can be modified accordingly.
- Under the situation of line-of-sight, we allow obstacles to reflect or deflect the wave. Reflection and deflection are not problems because each node reacts only to the first detected event. The only thing the system needs to maintain is an appropriate time interval between two successive localization events.
- We assume that background noise exists, and therefore band-pass filter can be used to listen to a particular wave frequency. This reduces the chances of false detection.

The event generation parameter here is the *source location of the event*. The distance between each node and the event source determines the rank of the node in corresponding node sequence. Figure 8 illustrates explanations and examples for the algorithms we proposed under the situation of wave propagation based events.

Figure 8 (a) shows an example for event generation parameter estimation using Algorithm 1. As shown in the figure, given anchor ordering (A C B), we know that wave propagation reached anchor A earlier than anchor C. Under the assumption of uniform propagation speed in all directions, for anchor pair ordering (A C), we can conclude that anchor A has shorter distance to the event source than anchor C. This indicates that the event source locates to right of dashed line l_1 which perpendicularly bisects the doted line connecting anchor A and anchor C. Similarly, for anchor pair ordering (C B), we can get that the event source locates to the left of dashed line l_2 which perpendicularly bisects the doted line connecting anchor C and anchor B. By extracting the joint area, we can get the estimated source location area of the event, shown as shaded area in the figure. From basic geometry knowledge, for $\triangle ACB$, three perpendicular bisection lines must joint at one point. So the combination of anchor pair ordering (A B) can not further contribute to parameter estimation, which is coincident with our analysis about *effective estimation unit* in Section 4.1. Actually, for wave propagation based events, each two consecutive anchors in the node sequence form an *effective estimation unit*.



(a) Event Source Location Estimation **(b) Segment Location Area Finding**

Figure 8. Wave Propagation Event Explanation

Figure 8 (b) shows a simple example for segment location area finding using Algorithm 2. For the first segment containing only normal node 3, the distance between event source and node 3 must be shorter than that between event source and anchor A. Investigating position point P_1, the event source needs to be located to the left of dashed line l_4 to satisfy the above requirement. P_1 is a possible location for node 3 because the required event source location area has overlapping part with the estimated event source location area. While for position point P_2, the event source needs to be located to the right of dashed line l_5. There is no overlapping with the estimated event source location area, therefore P_2 is not a possible position for node 3.

6 Overhead and Complicity Analysis

This section analyzes system overhead and computation complicity. In terms of system cost, we emphasize that our approach using uncontrolled events has two nice features:

- Localization using uncontrolled events eliminates the need for sophisticated event generation device. No control over either event generation or event distribution is required.
- The system adopts an asymmetric design in which resource constrained sensor nodes only need to detect and report events.

Those two features make the system deployment become quite flexible and convenient. In addition, the computation for localization is only about node sequences processing, which is supposed to be done outside of sensor nodes. We analyze the computation complicity on the node sequence processing side, where resources are plentiful.

From Algorithm 1, the computation complicity of event generation parameter estimation for one node sequence is $O(n)$, where n is the number of anchor nodes in the sequence. From Algorithm 2, the computation complicity of location area estimation for one segment in the node sequence is $O(S)$, where S is the size of the grid map. Algorithm 3, which integrates Algorithm 1 and Algorithm 2 and gives complete processing procedure for basic design, has a complicity of $O(m(O(n)+(n+1)(O(S)+i\cdot O(S)))+i\cdot O(S))$, namely:

$$O(m\cdot n\cdot i\cdot S) \tag{1}$$

Where m is the number of localization events, n is the number of anchor nodes, i is the number of normal nodes, and S is the size of the grid map.

Since the computation is done asymmetrically out of the resource constrained sensor node side, the computation complicity for the design in this paper is far from prohibitive.

7 Simulation Evaluation

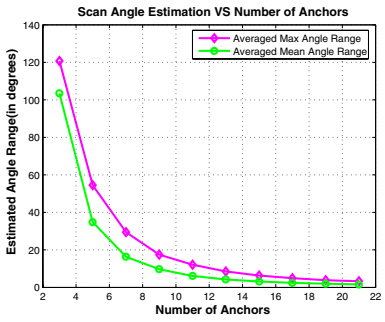
We evaluated the system design with both simulation and testbed implementation. In simulation, all the anchor nodes and normal nodes are deployed randomly with uniform distribution. All the simulations are based on the straight-line scan example and every event is generated with random scan angle. The statistics interested include (i) accuracy of event generation parameter estimation (mean and maximum range of $\widehat{\theta}$); (ii) ac-

curacy of normal node localization (mean and maximum localization error). All the statistics are averaged over 50 runs for high confidence. Table 1 illustrates the default simulation setup parameters.

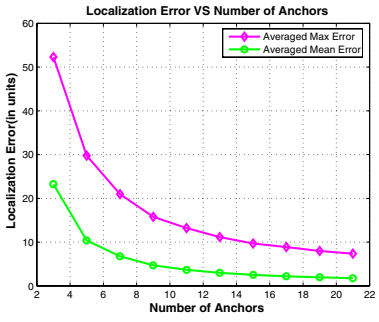
Table 1. Default Configuration Parameters

Parameter	Description
Field Area S	100×100
Event Type	Straight-line Scan
Number of Anchor Nodes	3 (Default)
Scan Times	6 (Default)
Number of Normal Nodes	100 (Default)
Random-Seed Loop	50 runs

Impact of Number of Anchors: In this experiment, we evaluate the scan angle estimation and localization error under a different number of anchors from 3 to 21 in steps of 2. The number of scans is 6 by default. Firstly, according to Section 4.1, for basic design, only the anchor nodes contribute to the event generation parameter estimation. Thus, we can expect that with more anchor nodes, more accurate estimation for the event generation parameter should be achieved. Figure 9(a) confirms our expectation. From Figure 9(a) we can see that, as the number of anchors increases, estimated angle range shrinks quickly. This means that with more anchors, more accurate scan angle estimation can be achieved. Actually, more anchor nodes also helps to divide the whole area into more small parts, as mentioned in Section 4.2, thus further benefits localization accuracy. Figure 9(b) shows that with more anchor nodes, localization error gets reduced significantly.



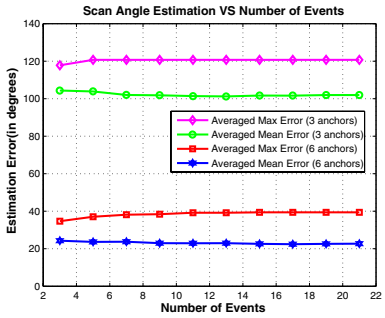
(a) Angle Range vs. Number of Anchors



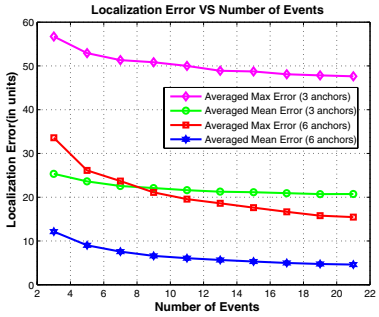
(b) Error vs. Number of Anchors

Figure 9. Impact of Number of Anchors

Impact of Number of Events: In this experiments, we investigated the impact of number of events. The system is setup up with 3 anchors and 6 anchors respectively. Because event generation parameter estimation is only affected by anchor nodes in the system, Figure 10(a) verifies that estimation accuracy keeps stable under different number of events. However, with more events, each normal node has more combination of location areas, thus smaller joint area is possible to be obtained. Figure 10(b) confirms this analysis. With increasing number of randomly generated events, the system error also gets reduced. We can notice from this figure that both using more anchors and accumulating more localization events can improve the system performance.



(a) Angle Range vs. Number of Events



(b) Error vs. Number of Events

Figure 10. Impact of Number of Events

The simulation in this subsections shows that (i) sensor node localization can be achieved using uncontrolled localization events; (ii) both number of anchors and number of localization events have impact on system performance.

8 System Evaluation

In this section, we report system implementation of our design on a physical testbed called Mirage, a large indoor testbed we built over a six-month period that can support up to 360 nodes. The whole testbed is composed of six 4-feet by 8-feet boards, illustrated in Figure 1. Each board in the system can be used as an independent sub-system, which is powered, controlled and metered separately. We use three high-end HITACHI CP-X1250 projectors, connected through a MATROX Triplehead2go graphics expansion box, to create a ultra-wide integrated display on six boards. Figure 1 shows that a long tilted line is generated through Mirage.

In our evaluation, totally 41 MICAz motes were mounted on the testbed as shown in Figure 1. We have implemented Java code for generating random straight-line scan at scanning line speed of 4.3 feet/s on Mirage testbed. We selected 6 nodes to be anchor nodes and the left 35 to be normal nodes. 10 random

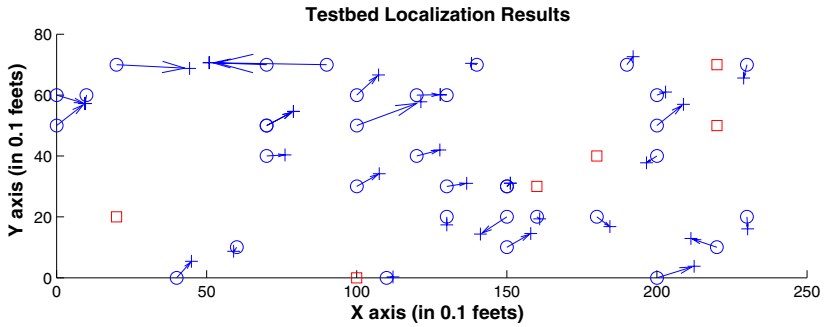


Figure 11. Testbed Localization Result Illustration

scans were conducted, thus 10 node sequences were obtained and processed. Figure 11 depicts the results. The whole area is modelled as a 240×80 grid map since the testbed has a size of 24 feet by 8 feet. In the figure, red squares stand for anchor nodes, and blue circles are the normal nodes. An arrow origins from the real position of each normal node and points to its estimated location (marked by a cross). From Figure 11, we can see that our algorithm successfully accomplished sensor node localization with totally uncontrolled scan events.

9 Conclusion

In this paper, we presented the first work exploiting uncontrolled events to localize sensor nodes. We demonstrate that normal sensor nodes can be localized in two steps: (i) firstly estimate the event generation parameter from the ordering of anchors in the node sequence; then (ii) compute the location area for each normal node according to its rank in the node sequences. We evaluated the system design at scale through analysis, extensive simulation, as well as testbed implementation. Results demonstrate that using only uncontrolled events, sensor node localization is achievable with great flexibility, low cost and good accuracy.

10 Acknowledgements

This research was supported in part by NSF grants CNS - 0626609 and CNS - 0626614.

11 References

[1] A. Baggio and K. Langendoen. Monte-Carlo Localization for Mobile Wireless Sensor Networks. In *MSN '06*.

[2] P. Bahl and V. N. Padmanabhan. Radar: An In-building RF-based User Location and Tracking System. In *InfoCom '00*.

[3] J. Bruck, J. Gao, and A. A. Jiang. Localization and Routing in Sensor Networks by Local Angle Information. In *MobiHoc '05*.

[4] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less Low Cost Outdoor Localization for Very Small Devices. *IEEE Personal Communications Magazine*, 7(4), August 2000.

[5] K. Chintalapudi, R. Govindan, G. Sukhatme, and A. Dhariwal. Ad-hoc Localization Using Ranging and Sectoring. In *InfoCom '04*.

[6] D. Culler, D. Estrin, and M. Srivastava. Overview of Sensor Networks. *IEEE Computer Magazine*, 37(8), 2004.

[7] M. Gaynor, S. Moulton, M. Welsh, A. Rowan, E. LaCombe, and J. Wynne. Wireless Sensor Network Applications. In *AMCIS '04*.

[8] D. K. Goldenberg, P. Bihler, M. Gao, J. Fang, B. D. O. Anderson, A. S. Morse, and Y. R. Yang. Localization in Sparse Networks using Sweeps. In *MobiCom '06*.

[9] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-free Localization Schemes in Large-scale Sensor Networks. In *MobiCom'03*.

[10] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, R. Stoleru, G. Zhou, Q. Cao, P. Vicaire, J. A. Stankovic, T. F. Abdelzaher, J. Hui, and B. Krogh. VigilNet: An Integrated Sensor Network System for Energy-efficient Surveillance. *ACM Transactions on Sensor Networks*, 2(1), 2006.

[11] X. Ji and H. Zha. Sensor Positioning in Wireless Ad-hoc Sensor Networks with Multidimensional Scaling. In *InfoCom '04*.

[12] B. Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *MobiCom '00*.

[13] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. Geographic Routing Made Practical. In *NSDI'05*.

[14] S. Kumar, T. H. Lai, and A. Arora. Barrier Coverage with Wireless Sensors. In *MobiCom'05*.

[15] loukas Lazos and R. Poovendran. SeRLoc: Secure Range-Independent Localization for Wireless Sensor Networks. In *WiSe '04*.

[16] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust Distributed Network Localization with Noisy Range Measurements. In *SenSys '04*.

[17] A. Nasipuri and R. E. Najjar. Experimental Evaluation of an Angle Based Indoor Localization System. In *WiNMee '06*.

[18] D. Niculescu and B. Nath. Ad Hoc Positioning System (APS) using AOA. In *InfoCom '03*.

[19] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-support System. In *MobiCom '00*.

[20] K. Römer. The Lighthouse Location System for Smart Dust. In *MobiSys '03*.

[21] A. Savvides, C. C. Han, and M. B. Strivastava. Dynamic Fine-grained Localization in Ad-hoc Networks of Sensors. In *MobiCom '01*.

[22] Y. Shang, W. Ruml, Y. Zhang, and M. P.J.Fromherz. Localization from Mere Connectivity. In *MobiHoc '03*.

[23] G. Simon, M. Maróti, Ákos Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton. Sensor Network-based Countersniper System. In *SenSys '04*.

[24] R. Stoleru, T. He, J. A. Stankovic, and D. Luebke. A High-accuracy, Low-cost Localization System for Wireless Sensor Networks. In *Sensys'05*.

[25] R. Stoleru, P. Vicaire, T. He, and J. A. Stankovic. Stardust: A Flexible Architecture for Passive Localization in Wireless Sensor Networks. In *Sensys'06*.

[26] A. Terzis, A. Anandarajah, K. Morre, and I.-J. Wang. Slip Surface Localization in Wireless Sensor Networks for Landslide Prediction. In *IPSN '06*.

[27] E. W. Weisstein. Plane Division by Lines. *mathworld.wolfram.com*.

[28] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh. Monitoring Volcanic Eruptions with a Wireless Sensor Network. In *EWSN '05*.

[29] C. D. Whitehouse. The Design of Calamari: an Ad-hoc Localization System for Sensor Networks. *Master Degree Project Report, University of California at Berkeley*, 2002.

[30] K. Whitehouse, C. Karlof, and D. Culler. A Practical Evaluation of Radio Signal Strength for Ranging-based Localization. *SIGMOBILE Mob. Comput. Commun. Rev.*, 11(1), 2007.

[31] K. Whitehouse, C. Karlof, A. Woo, F. Jiang, and D. Culler. The Effects of Ranging Noise on Multihop Localization: An Empirical Study. In *IPSN '05*.

[32] K. Yedavalli, B. Krishnamachari, S. Ravula, and B. Srinivasan. Ecolocation: A Sequence Based Technique for RF-only Localization in Wireless Sensor Networks. In *IPSN '05*.

[33] Z. Zhong and T. He. MSP: Multi-Sequence Positioning of Wireless Sensor Nodes. In *Sensys'07*.