

SecMCL: A Secure Monte Carlo Localization Algorithm for Mobile Sensor Networks

Yingpei Zeng[†] Jiannong Cao[‡]

[†]State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, P.R. China
Email: {zyp,jhong,zsg}@dislab.nju.edu.cn

Jue Hong[†] Shigeng Zhang[†] Li Xie[†]

[‡]Department of Computing
Hong Kong Polytechnic University
Hong Kong
Email: csjcao@comp.polyu.edu.hk, xieli@nju.edu.cn

Abstract

Recently with the emergence of mobile sensor networks, localization for such networks has gained much attention, and many localization algorithms have been proposed. Among them the Sequential Monte Carlo (SMC) based algorithms are very popular because of their simplicity and efficiency. However, most current SMC-based localization algorithms implicitly assume there is no attacker in the network, which may not be true in real applications. The attackers, if any, may send false information by themselves or through compromised nodes to disturb the localization. In this paper, we present the design and evaluation of a Secure Monte Carlo Localization algorithm, SecMCL. SecMCL provides authentication to messages and employs a new sampling method to defeat attacks. Simulation results show that SecMCL greatly improves the localization accuracy of existing SMC-based localization method when there are attacks. Also, compared with existing SMC localization method, SecMCL incurs no communication cost (in terms of number of messages) and achieves the same localization accuracy when there is no attack.

1. Introduction

Localization is important in wireless sensor networks since many applications and supporting techniques of the networks require nodes to know their locations. E.g., applications such as habitat monitoring [1], wildlife tracking [2], vehicle tracking [3], and volcano monitoring [4], as well as supporting techniques such as geographic routing [5] and geographic key distribution [6]. A straightforward solution is to equip each node with a GPS receiver; however it is too costly and only affordable in small networks.

Many localization algorithms have been proposed to enable nodes to estimate their locations in static networks [7]–[11], assuming partial nodes (called beacons) know their locations (e.g., equipped with GPS receivers). However, when these algorithms are used in mobile sensor networks (e.g., [2]), they suffer from high overhead and low performance

[12], [13], because they need to periodically restart the whole localization process. When the mobility of the network is high, the collected information may even already be stale before being fed to the localization algorithm.

The Sequential Monte Carlo (SMC) based localization algorithms, specially designed for mobile sensor networks, are very popular in the research community [12]–[18]. They are simple to implement [12], [13] and can even utilize the mobility of nodes for localization. The key idea of SMC-based localization is to represent the posterior distribution of node's location by a set of weighted samples, and in each time unit, the samples are updated based on new observations about beacons.

However, almost all the current SMC-based localization algorithms do not consider the attackers in the networks. In reality, the attackers may try to distort nodes' location estimations to make the localization failed. They may directly inject false messages into the network or let the compromised nodes broadcast false information. Many secure localization algorithms have been proposed [19], but none of them can be used for SMC-based localization.

In this paper, we present the design and evaluation of a Secure Monte Carlo Localization algorithm, called SecMCL. SecMCL follows the SMC localization framework [13], however, it authenticates the messages and executes a new sampling algorithm when it finds abnormality (i.e., cannot obtain enough valid samples). Simulation results show that the localization accuracy of SecMCL is much higher than existing algorithms under attacks. Also, compared with existing SMC localization algorithms, SecMCL does not need additional messages (sending and receiving messages are the most energy-consuming operations in sensor node [20]), and achieves the same localization accuracy when there is no attack (which makes it easier to be adopted by network operators).

The rest of the paper is organized as follows. In Section 2 we review related work. In Section 3, we describe the network and attacker models. In Section 4, we present the design of SecMCL. Section 5 gives simulation results of SecMCL. Finally we conclude this paper in Section 6.

2. Related Work

Many localization algorithms have been proposed for mobile sensor networks [12]–[18], [21], [22] since traditional localization algorithms are inefficient in mobile environments [12], [13]. Most of these algorithms use the SMC method [12]–[18], which can naturally take advantages of nodes’ mobility to improve accuracy. Other methods like periodically restarting traditional localization algorithms [21], [22] have high energy consumption or need additional hardware (e.g., accelerometer [22]).

We briefly review the SMC-based localization algorithms here. MCL [13] first proposed to use SMC [23] for localization in mobile sensor networks. In MCL, each node’s location is represented by a set of weighted samples. In each time unit, nodes predict new samples based on samples in previous time unit, and filter out invalid samples based on new observations about 1-hop and 2-hop beacons. MCB [15] reduced the time spent in sampling in MCL. It computes a bounding box first, and then samples in the box to get valid samples with higher probability. MMCL [17] used the observations about all beacons at the same time, but not only beacons within two hops. It needs to periodically flood the beacon messages thus has high communication cost. MSL*, MSL [18] and the algorithm in [12] improved MCL by combing the information of common nodes within two hops when filtering samples. However, they need additional messages to pass sample or accuracy information. In [14] and [16], the authors used SMC method for the cases that range or angle-of-arrival (AoA) measurements are available.

All the mentioned SMC-based localization algorithms do not consider the adversaries in the network. Although in [13], the authors gave a section to discuss security, they did not present a clear solution. Many secure localization algorithms have been proposed [19]; however they all are based on traditional localization algorithms, e.g., LMS by Li et al. employs traditional DV-hop method to get distance measurements in multihop networks [19]. Thus they are inefficient in mobile sensor networks [12], [13]. In contrast, our new secure localization algorithm presented in this paper can even take advantages of network mobility.

3. Network Model and Attacker Model

Network model: The network consists of a set of common nodes and a set of beacon nodes. Beacons can always know their locations through equipped GPS receivers. Both the common nodes and the beacons may be mobile. The maximum speed of all the nodes is v_{max} . The transmission range of all the nodes is r . We assume each node a has a private key K_a^{-1} which is used to sign messages. Also other nodes are able to verify signatures. Several public-key libraries for sensor networks are available now [24].

Attacker model: We consider *external* and *internal* attacks. In the external attacks, the adversaries do not have

the security credentials of sensor nodes. However they can still inject false messages into the network. Such attacks can be easily defeated by message authentication. In the internal attacks, the adversaries have compromised a small number of nodes (beacons or common nodes). Thus they can let the compromised nodes stop sending messages, or even worse, send false but authenticated messages. Specifically, we assume the compromised beacons may send false locations in the HELLO packets, and the compromised common nodes may forward the location claims received from common nodes (we call the behavior as *malicious relaying*, since normal nodes only forward the location claims received directly from beacons as explained later). Currently we do not consider the defense of wormhole attacks [25] here.

4. SecMCL: A Secure Monte Carlo Localization Algorithm

In this section, we present the design of SecMCL, which includes the design of communication protocol and the design of localization algorithm. Same as MCL [13], SecMCL only uses the observations on 1-hop and 2-hop beacons. Different from some SMC-based algorithms [12], [17], [18], we do not use the information of common neighbor nodes or 2-hop-away beacons, because 1) obtaining such information requires extra messages [12], [17], [18] and 2) the improvement on localization accuracy is limited in some cases, e.g., improvement is less than $0.1r$ in figures 12, 13 of [12] when $v_{max} > 0.4r$ or $s_d > 2$ (s_d is the seed/beacon density).

4.1. Communication Protocol

Similar to previous algorithms [12]–[18], we assume time is divided into discrete time units. In each time unit, nodes need to get their current locations since they moved in the last time unit. Before running the localization algorithm, nodes will communicate to get the required information as shown below.

Communications in SecMCL

Beacon B: $claim = sig_{K_B^{-1}}(ID_B, loc_t, t)$

Beacon B \rightarrow *: $sig_{K_B^{-1}}(HELLO, ID_B, claim)$

Node N \rightarrow *: $sig_{K_N^{-1}}(HELLO, ID_N, \{claim_i\})$

First, the beacons sign messages containing their location claims and broadcast. The location claim consists of node id (ID_B), current location (loc_t), and current time (t). The location claim is also signed to ensure others cannot generate fake claims. Second, the common nodes sign messages containing the location claims received from their neighbor beacons and broadcast. Then the communication is end. After this phase, all the common nodes know their 1-hop and 2-hop neighbor beacons. The number of messages used is the same with MCL [13].

4.2. Location Estimation Algorithm

Our location estimation algorithm consists of two processes: a process same as the MCL [13] and a new sampling process. If the first process ends without abnormality (i.e., get enough samples), the second process is skipped, otherwise the second process is started.

SecMCL first executes the MCL process. Following the SMC method [23], MCL [13] running at each node contains 3 steps: 1) the initialization step, which is to randomly draw N samples in the deployment area to a set of samples, L_0 , 2) the prediction step, which is to predict new samples based on previous set of samples L_{t-1} and the transition equation

$$p(l_t|l_{t-1}) = \begin{cases} \frac{1}{\pi v_{max}^2} & \text{if } d(l_t, l_{t-1}) < v_{max} \\ 0 & \text{if } d(l_t, l_{t-1}) \geq v_{max}, \end{cases} \quad (1)$$

where l_{t-1} is a sample in L_{t-1} , and $d(l_t, l_{t-1})$ is the distance between two samples, and 3) the filtering step, which is to discard the samples that do not satisfy the filter condition, for any sample l

$$filter(l) = \forall s \in S, d(l, s) \leq r \wedge \forall s \in T, r < d(l, s) \leq 2r, \quad (2)$$

where S and T are the 1-hop neighbor beacons and 2-hop neighbor beacons of the node respectively. The prediction and filtering steps are repeated, until N valid samples have been acquired.

When there are no attacks, MCL can always draw enough samples, because sampling near the real location of the node can always get valid samples. However, when the adversaries have compromised part of nodes, the predication and filtering steps may fail to get enough valid samples. The compromised beacons may broadcast wrong locations, e.g., a beacon at location P claims its location is P' in the HELLO packet. Also, the compromised common nodes may maliciously relay location claims pretending to be 1-hop neighbors of corresponding beacons. For example, node N_a is not a 1-hop neighbor of beacon B , but when it hears B 's location claim from N_b , it adds B 's location claim into its HELLO packet. We discuss the effects of such attacks on the sampling and try to learn the clue to defend against them:

- 1) When the sample set L_{t-1} in the previous time unit is correct, as shown in Figure1(a), then the samples generated in current predication step is also correct (reflecting real location), and the predicted samples will intersect with the *filter regions*¹ of beacons which are not compromised or maliciously relayed (i.e., $B1$ and $B2$). However, the compromised beacon $B1'$ (maliciously relayed beacon is similar) in Figure1(a) makes all the samples fail to satisfy filter condition (2).

1. The filter region of a 1-hop beacon is a *circle* centering at beacon's location with radius r , and the filter region of a 2-hop beacon is a *ring* centering at beacon's location with inner radius r and outer radius $2r$.

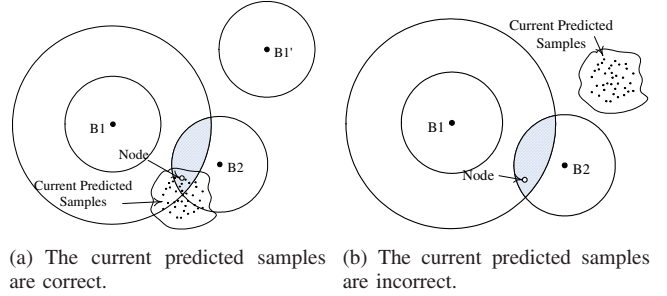


Figure 1. MCL's sampling under attacks.

- 2) When the sample set L_{t-1} in the previous time unit is already incorrect (e.g., the node only heard one compromised beacon in the $t-1$ time unit and is misled), the samples generated in current predication step is also incorrect, and as shown in Figure1(b), all the samples fail to satisfy filter condition (2) too.

From the discussion we can see that in the first case the effect of compromised beacons (e.g., $B1'$) should be erased, and in the second case the effect of incorrect sample set should be erased. Our second process is exactly to carry out such "erasion". It's a new secure sampling method which runs when not enough samples are obtained after running MCL. The whole process is shown in Algorithm 1. It contains two phases. *In the first phase*, we still predicate samples based on the previous sample set L_{t-1} , and consider the new samples as valid if it is in the intersection of the filter regions of i beacons (any i beacons). Then the new filter condition for sample l becomes

$$filter(l, i) = \exists Q(Q \subseteq S \cup T) \wedge (size(Q) = i) \wedge (\forall s(s \in S \wedge s \in Q \rightarrow d(l, s) \leq r)) \wedge (\forall s(s \in T \wedge s \in Q \rightarrow r < d(l, s) \leq 2r)) \quad (3)$$

where $size(X)$ is to compute the cardinality of a set X^2 . We try $i = size(S \cup T) - 1$ first. If we still cannot obtain enough valid samples, we will reduce i by one and repeat such sampling until i reaches 1. If SecMCL still cannot get enough samples in the first phase, it will enter the second phase. *In the second phase*, SecMCL directly draws new samples from the intersection of filter regions of any i beacons (i.e., ignore the previous sample set). Similarly, we try $i = size(S \cup T) - 1$ first and repeat until $i = 1$.

4.3. Analysis

Resolution limit: In [10], a theoretical limit on resolution is proposed, which is the distance a sensor can move without changing the connectivity. Following the analysis, if we

2. Inspired by MCB [15], our implementation of filtering also calculates the plausible sampling region first. However, in our algorithm representing the plausible sampling region by a single box is not efficient (too large); we divide the minimum rectangle covering all beacons' filter regions into grids and select the grids intersecting with more than i beacons' filtering regions as plausible sampling regions.

Algorithm 1 New sampling method in SecMCL.

Input: S, T, N , and L_{t-1} .

```
1:  $n = \text{size}(S \cup T)$ . //compute the number of beacons.
2: for  $i = n - 1$  to 1 do
3:   Predicate samples based on the transition equation (1).
4:   Filter the samples using filter condition (3).
5:   if number of samples  $\geq N$  then
6:     Break.
7:   end if
8: end for
9: for  $i = n - 1$  to 1 do
10:  Randomly generate samples in the deployment area.
11:  Filter the samples using filter condition (3).
12:  if number of samples  $\geq N$  then
13:    Break.
14:  end if
15: end for
```

use beacons within two hops the resolution limit is $\frac{\pi r}{12s_d}$ [13], where s_d is the seed density (the average number of seeds/beacons within one hop). In our scenario, we denote the densities of good and compromised beacons by s'_d and s''_d respectively ($s'_d > s''_d$). Then the resolution limit is also $\frac{\pi r}{12s'_d}$, since SecMCL filters the impact of compromised beacons in general cases.

Security analysis: We discuss attacks according to the attacker model in Section 3. *First*, the packets injected by the adversaries cannot pass the authentication. *Second*, if the compromised nodes stop communicating during localization, the impact is limited because small variation of node density is tolerable when density is enough [13]. *Third*, when the compromised beacons broadcast false locations and compromised common nodes maliciously relay location claims, the secure sampling method of SecMCL can filter them, or in the worst case, bound the error caused by them. We use n_g and n_k to denote the numbers of good and compromised beacons heard respectively ($n_g > n_k$). For the page limit we only analyze the scenario with a correct previous sample set. There are two cases: 1) the adversaries trying to mislead the node to a location not within the filter regions of any good beacons, and 2) the adversaries trying to mislead the node to a location within the filter regions of some good beacons (n_c). In the first case, the attack will fail because SecMCL takes samples from the regions as maximal as possible covered by the filter regions of beacons, and even if the compromised beacons collude, the number of filter regions covering the false location (n_k) is less than the number of filter regions covering the node's real location (n_g). In the second case, the adversaries utilize partial good beacons (n_c) similar to the pollution attack³ [26]. Then if the adversaries want to succeed, n_c should be greater than $n_g - n_k + 1$ (i.e., the false location has $n_g + 1$ filter regions covering it), and they can only mislead a small number of

3. The success of such attack requires: the adversaries know the locations of the node and good beacons, and the adversaries are colluding.

Table 1. Parameters for simulations

Symbol	Default value	Meaning
v_{max}	0.4r	maximum speed
n_d	10	average number of nodes within one hop
s_d	2	average number of seeds within one hop
r_b	7.8%	ratio of compromised beacons to total beacons
r_n	0%	ratio of compromised common nodes to total common nodes
dm_{max}	2r	maximum distance between the claimed location and real location (for compromised beacons)
doi	0	degree of irregularity [11]

nodes that heard the same beacons. Thus, in the worst case the localization error is still bounded by the intersection of the filter regions of $n_g - n_k + 1$ good beacons.

5. Evaluation

We get the simulator developed by Hu and Evans [13] and implement SecMCL in it (our code can also be acquired by email). For all of our experiments, sensor nodes are randomly distributed in a 500m×500m region. r is set to 50m. We use the modified random waypoint model [13]. The parameters we vary are shown in Table 1⁴. We compare SecMCL with other three algorithms: MCL [13], Centroid [8] and Amorphous [10]⁵. The performance of an algorithm is judged by the average location estimation error of all the common nodes. All the results we present here are at least the average of five independent runs.

Accuracy: Figure 2 shows the estimation error of different algorithms from time unit 0 to time unit 50. The ratio of compromised beacons is 7.8% (5 beacons). Each compromised beacon randomly selects a false location (0, dm_{max}] away from its real location to broadcast. We can see that both SecMCL and MCL are more accurate than Centroid and Amorphous since they can naturally use the historical location information. The estimation error of SecMCL is the lowest (0.5r lower than MCL), because it filters the effect of the compromised beacons. Also SecMCL converges faster than MCL due to its filtering ability.

Compromised beacon ratio: Figure 3 shows the estimation error when we vary r_b . We can see that when there is no attack (i.e., $r_b = 0$), SecMCL has the same estimation error with MCL since the secure sampling process is skipped. Although the estimation error of all the algorithms increase with r_b , SecMCL outperforms others in the localization accuracy. This figure also indicates that MCL is most affected

4. The default value of s_d is set to 2 but not 1 [13] for varying the number of compromised beacons over a larger range. The default value of v_{max} is set to 0.4r but not 0.2r per time unit [13], because MCL has the best performance at that speed [12], [13].

5. Note that Amorphous [10] has much more communication overhead than other three algorithms which we do not show in our simulations.

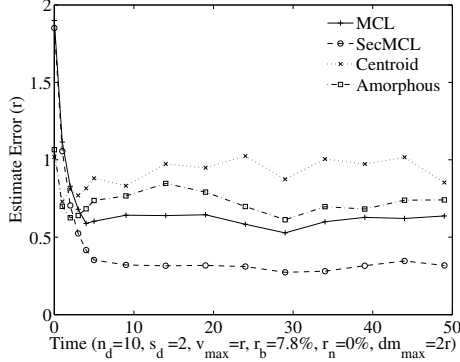


Figure 2. Accuracy comparison.

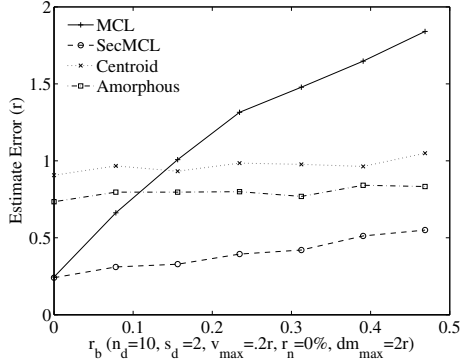


Figure 3. Impact of compromised beacon ratio.

by compromised beacons, because it usually fails to get valid samples even with one compromised beacon.

Distortion distance: Figure 4(a) shows the impact of different dm_{max} . The estimation error of all the algorithms increases with dm_{max} , which is very natural. When dm_{max} is smaller than $0.5r$, MCL and SecMCL have similar estimation error because the filter regions of compromised beacons may still cover the real location and then secure sampling process of SecMCL is skipped; however, when $dm_{max} > r$ the estimation error of MCL increases quickly because MCL already cannot get any valid sample even with single compromised beacon then, e.g., in Figure 1(a).

Compromised common node ratio: Figure 4(b) shows the impact of different r_n . Compromised common nodes will maliciously relay the location claims as mentioned. However, if the compromised common nodes forward all the heard location claims, the number of claims forwarded by each compromised node will be $4s_d$, 4 times of the number of claims forwarded by a normal node (s_d). Thus here we limit the number of maliciously relayed location claims to be s_d (i.e., total $2s_d$ claims). We can see that the estimation error of SecMCL is still less than $0.5r$ even when r_n is over 50%. We do not simulate Amorphous here because it does not make sense: compromised nodes can launch more serious attacks such as reducing the hop count [27].

Speed: Figure 4(c) shows the location estimation error

when we vary the speed of nodes. Both MCL and SecMCL perform best when $v_{max} = 0.4r$ per time unit, and then their performance degrades slightly when v_{max} increases. SecMCL also has the lowest location estimation error because of its filtering ability under attacks.

Seed density: Figure 4(d) shows the location estimation error of different algorithms when seed density varies. It is natural that when the seed density increases the estimation error of all the algorithms decreases. The estimation error of Amorphous does not have much variation after s_d reach 1, similar to the finding in [13]. SecMCL performs better than other algorithms when s_d is enough (i.e., ≥ 1).

Node density: Figure 4(e) shows the impact of node density. The estimation error of Amorphous decreases obviously when node density increases, because the estimation of one-hop distance becomes more accurate. SecMCL and Centroid are little affected by node density because they only need a small number of neighbors to get the information of beacons. A surprising finding is the estimation error of MCL increases slightly when node density increases. The reason may be the false beacon information is distributed to nodes within two hops more sufficiently then.

Radio irregularity: Figure 4(f) shows the impact of degree of irregularity (DOI) [11] in different algorithms. DOI is a method to approximate the irregularity of radio range, e.g., when DOI is 0.1, then the actual radio range in each direction is randomly chosen from $[0.9r, 1.1r]$. Since SecMCL can filter partial irregularity as compromised beacons, the estimation error of SecMCL is still small when $DOI \leq 0.3$, which is better than MCL and MCB and the algorithm in [12] (their performance do not degrade only when $DOI \leq 0.2$). Also, MCL performs badly when $DOI \geq 0.3$. The reason is the radio irregularity together with compromised beacons makes MCL harder to get valid samples.

6. Conclusion

SMC-based localization is popular in mobile sensor networks, however, nearly all the existing SMC-based localization algorithms do not consider security issues. To the best of our knowledge, this is the first work to study secure SMC-based localization. Our SecMCL method prevents the adversaries from injecting false messages by authentication, and can tolerate compromised nodes by a new sampling method. Simulation results show that the localization accuracy of SecMCL is much higher than MCL under attacks. SecMCL also has two merits compared with some secure localization algorithms [19]. First, SecMCL does not incur additional messages, which conserves much energy. Second, when there is no attack, SecMCL achieves the same localization accuracy as MCL, which makes it also suitable for networks currently operating in trusted environment.

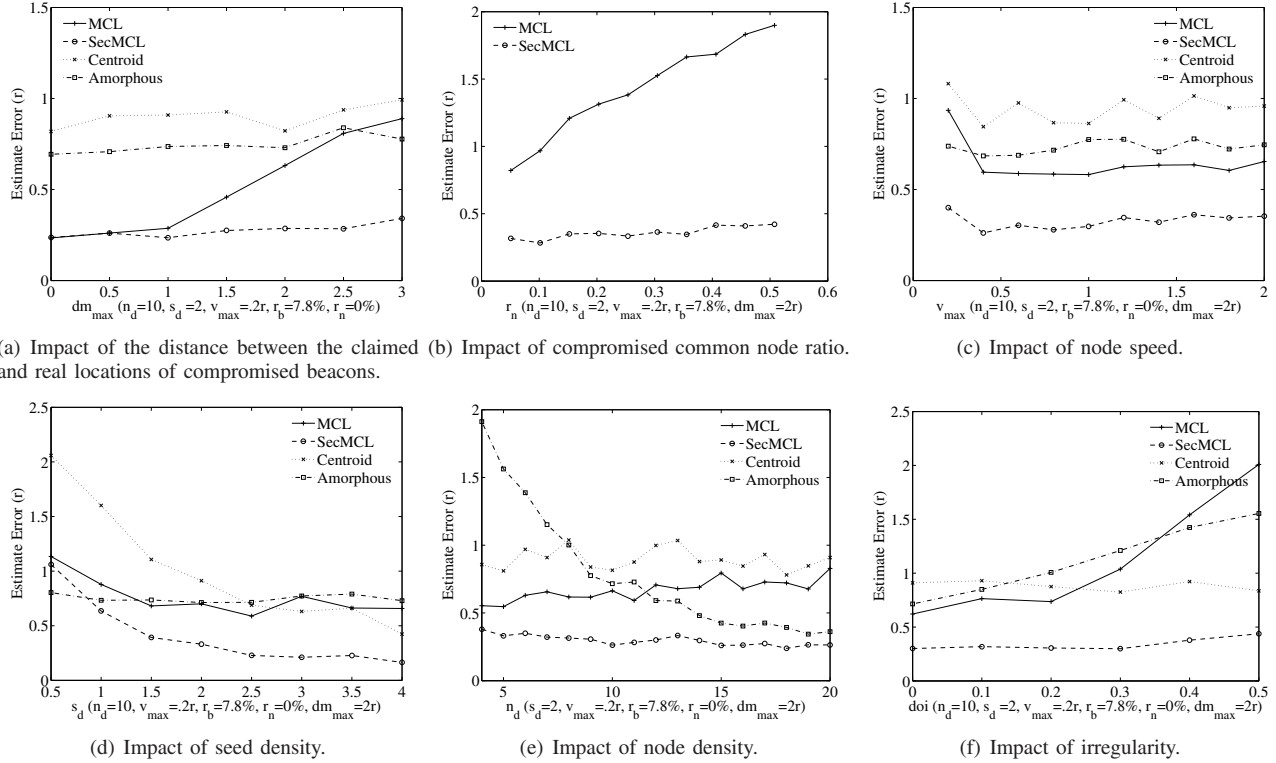


Figure 4. Varying dm_{max} , r_n , v_{max} , s_d , n_d and doi in simulation.

Acknowledgment

This work is supported in part by Hong Kong Research Grant Council under CERG grant PolyU 5102/07E, the Hong Kong Polytechnic University under the ICRG grant G-YF61, Natural Science Foundation of China under Grants No.60673154, No.60573132, No.60873026, and No.60573106, China 973 Project under Grant No. 2006CB303000, and Natural Science Foundation of Jiangsu Province under Grant “Research and Realization on ASLR in operating systems”. The authors are also grateful to Hu and Evans for sending us their code of [13].

References

- [1] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, “Habitat monitoring with sensor networks,” *Communications of the ACM*, vol. 47, no. 6, pp. 34–40, 2004.
- [2] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, “Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet,” in *Proceedings of ASPLOS*, 2002, pp. 96–107.
- [3] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh, “Energy-efficient surveillance system using wireless sensor networks,” in *Proceedings of MobiSys*, 2004, pp. 270–283.
- [4] W.-Z. Song, R. LaHusen, R. Huang, A. Ma, M. Xu, and B. Shirazi, “Airdropped sensor network for real-time high-fidelity volcano monitoring,” in *Proceedings of MobiSys*, 2009.
- [5] B. Karp and H. T. Kung, “GPSR: greedy perimeter stateless routing for wireless networks,” in *Proceedings of MobiCom*, 2000.
- [6] D. Liu and P. Ning, “Location-based pairwise key establishments for static sensor networks,” in *Proceedings of ACM SASN*, 2003.
- [7] G. Mao, B. Fidan, and B. D. O. Anderson, “Wireless sensor network localization techniques,” *Computer Networks*, vol. 51, no. 10, pp. 2529–2553, 2007.
- [8] N. Bulusu, J. Heidemann, and D. Estrin, “Gps-less low-cost outdoor localization for very small devices,” *IEEE Personal Commun. Mag.*, vol. 7, no. 5, pp. 28–34, 2000.
- [9] D. Niculescu and B. Nath, “Ad hoc positioning system (APS),” in *Proceedings of IEEE GLOBECOM*, 2001.
- [10] R. Nagpal, H. Shrobe, and J. Bachrach, “Organizing a global coordinate system from local information on an ad hoc sensor network,” in *Proceedings of IPSN*, 2003.
- [11] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, “Range-free localization schemes for large scale sensor networks,” in *Proceedings of MobiCom*, 2003, pp. 81–95.
- [12] S. Zhang, J. Cao, L. Chen, and D. Chen, “Locating nodes in mobile sensor networks more accurately and faster,” in *Proceedings of SECON*, June 2008, pp. 37–45.
- [13] L. Hu and D. Evans, “Localization for mobile sensor networks,” in *Proceedings of MobiCom*, 2004, pp. 45–57.
- [14] B. Dil, S. O. Dulman, and P. J. M. Havinga, “Range-based localization in mobile sensor networks,” in *Proceedings of EWSN*, 2006, pp. 164–179.
- [15] A. Baggio and K. Langendoen, “Monte carlo localization for mobile wireless sensor networks,” in *Proceedings of MSN*, 2006.
- [16] R. Huang and G. Zaruba, “Incorporating data from multiple sensors for localizing nodes in mobile ad hoc networks,” *IEEE Trans. Mobile Comput.*, vol. 6, no. 9, pp. 1090–1104, Sept. 2007.
- [17] J. Yi, S. Yang, and H. Cha, “Multi-hop-based monte carlo localization for mobile sensor networks,” in *Proceedings of SECON*, 2007, pp. 162–171.
- [18] M. Rudafshani and S. Datta, “Localization in wireless sensor networks,” in *Proceedings of IPSN*, 2007, pp. 51–60.
- [19] A. Boukerche, H. Oliveira, E. Nakamura, and A. Loureiro, “Secure localization algorithms for wireless sensor networks,” *IEEE Commun. Mag.*, vol. 46, no. 4, pp. 96–101, 2008.
- [20] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S. J. Pister, “System architecture directions for networked sensors,” in *Proceedings of ASPLOS*, 2000, pp. 93–104.
- [21] S. Tilak, V. Kolar, N. B. Abu-Ghazaleh, and K.-D. Kang, “Dynamic localization protocols for mobile sensor networks,” in *Proceedings of MASS*, 2004.
- [22] T.-H. Lin, P. Huang, H.-H. Chu, and C.-W. You, “Energy-efficient boundary detection for rf-based localization systems,” *IEEE Trans. Mobile Comput.*, vol. 8, no. 1, pp. 29–40, 2009.
- [23] A. Doucet, N. Defreitas, and N. Gordon, *An Introduction to Sequential Monte Carlo Methods*. New York: Springer-Verlag, 2001. [Online]. Available: http://www.cs.ubc.ca/~arnaud/doucet_defreitas_gordon_smcbookintro.ps
- [24] A. Liu and P. Ning, “TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks,” in *Proceedings of IPSN*, 2008, pp. 245–256.
- [25] Y. Hu, A. Perrig, and D. Johnson, “Packet leashes: A defense against wormhole attacks in wireless ad hoc networks,” in *Proceedings of INFOCOM*, 2003.
- [26] Y. Zeng, J. Cao, S. Zhang, S. Guo, and L. Xie, “Pollution attack: A new attack against localization in wireless sensor networks,” in *Proceedings of WCNC*, 2009.
- [27] Y. Zeng, S. Zhang, S. Guo, and X. Li, “Secure hop-count based localization in wireless sensor networks,” in *Proceedings of CIS*, 2007.