

A Variation on Hop-counting for Geographic Routing

Yurong Xu

James Ford

Fillia S. Makedon

Dept. of Computer Science and Dartmouth Experimental Visualization Laboratory (DEVLAB),
Dartmouth College, Hanover NH 03755, USA
{yurong, jford, makedon}@cs.dartmouth.edu,

Abstract: In this paper, we propose a new abstraction for localization — Hop Coordinates — that can improve the accuracy of geographic routing over that possible with a localization algorithm based only on hop counting. As compared with previous proposed localization algorithms, which require additional hardware or special antennas, our technique is similar to hop counting and only requires the connectivity of network. Using simulations, we show that incorporating our technique improves the accuracy of geographic routing in localization results generated by MDS-MAP and MDS-MAP(P) by 22% and 14%, respectively.

1. Introduction

Wireless Sensor Networks (WSNs) [1], an emerging new type of ad-hoc network, integrate sensing, processing and wireless communication in a distributed system. WSNs have numerous applications, such as surveillance, healthcare, industry automation, and military uses.

Geographic routing is an important routing scheme in WSNs [2]. It requires every node in a WSN to be aware of its location as accurately as possible so that routing can be based on positional information instead of logical addresses.

There are many physical features in WSN that have been utilized by different localization algorithms to generate the location of nodes in a WSN: the angle of arrival (AOA) [17], time of arrival (TOA) [19], time difference of arrival (TDOA) [22], Receive Signal Strength Indicator (RSSI) [8] and hop information and connectivity. Usually, sensor nodes must deploy additional hardware for AOA, TOA and TDOA, or special antennas for AOA. As to RSSI, the utilization of RSSI depends on a rough relationship between the distance and the signal strength, but this relationship is affected unpredictably [8] by the antenna, battery, electric components inside of the node and the outside environment. In contrast with other physical features which we can use in localization algorithms, hop information and connectivity are much more stable and can be achieved in most currently deployed WSNs without any additional hardware.

Extensive simulation experiments on different localization algorithms [9,10,14-16] based on only

hop-counting or connectivity have shown that there is usually a sizeable error in the location generated by the algorithms in comparison with actual locations, even after some refinement processes such as the ones in [15] and [16]. Such errors lead to the high possibility of non-optimal geographic routing.

This paper does not propose a completely new localization algorithm; rather, it introduces a novel abstraction called *hop coordinates*, which not only counts the number of hops during routing, but also offsets this total based on the local network structure.

When combined with the localization algorithms MDS-MAP [15] and MDS-MAP(P) [16], a hop coordinates-based technique achieves about a 22% and 14% improvement, respectively, in routing accuracy using the AODV routing algorithm in NS-2, comparing with results without hop coordinates.

2. Related Work

Node localization in WSNs has been an active research field for some time, and many localization algorithms have been proposed. Based on the different physical features of WSNs, we can roughly separate these into several categories: some algorithms, such as [3], [9], [10], [14], [15] and [16], merely utilize the connectivity or hop count information in a WSN; other algorithms, such as [17], uses AOA in addition to hop-counting; some, such as [19], uses TOA information, and some (e.g. [22]) uses TDOA. Hop information and connectivity are available in currently implemented WSNs, while AOA requires additional devices such as ultrasound receivers [13] or directional antennas, and TOA and TDOA require nodes to have two different propagation speed communication devices such as ultrasound transceivers [13] and RF transceivers to measure the delivery time. Recently, [20] proposed Spotlight, a light-based localization system that lies on additional hardware, while [21] proposed a novel method that doesn't need additional hardware but does require anchor nodes.

In the remainder of this paper, we will only consider localization algorithms *that use only hop information and connectivity*, and that thus need no additional hardware and no special nodes such as anchor nodes. This will exclude the algorithms such as BVR[18]. But this still leaves us with several

algorithms. DV-Hop, DV-distance by Niculescu and Nath [9], and Hop-TERRAIN by Savarese et al. [10] use hop-counting to calculate the location of nodes. However, these methods are outperformed by the following MDS-MAP series algorithms.

Based on connectivity in WSNs, Shang et al. presented a novel algorithm called MDS-MAP(P) [16], which extends MDS-MAP [15] into a distributed algorithm. MDS-MAP(P) first computes local maps for each node with MDS using local shortest paths, then merges local maps into a global map.

Though the main idea of hop coordinates derives from hop-counting, our technique can not only be applied directly in the localization algorithms based on hop-counting, but can also be applied in the localization algorithms which need connectivity, such as MDS-MAP series algorithms. In our simulation experiments in this work, we apply hop coordinates in MDS-MAP series algorithms.

3. Hop Coordinates

3.1 Definition of Hop Coordinate

Unlike other localization algorithms, which only count number of hops or utilize connectivity between nodes, the proposed *Hop coordinates* abstraction introduced in this paper not only counts the number of hops from some bootstrap node, but also offsets it with the local network information of that node.

Definition 1: A *Hop Coordinate* is constructed from two parts: *number of hops* and *offset*. The first part is a positive integer which equals the number of hops in a minimum hop route from some bootstrap node to a given node. The second part can be seen as a decimal fraction generated from local network information, as defined as:

$$offset_A = \begin{cases} \frac{\sum_{B \in N_A} (hop_B - (hop_A - 1)) + 1}{2(|N_A| + 1)}, & \text{when } |N_A| \neq 0; \\ 0, & \text{when } |N_A| = 0. \end{cases}$$

(here, A, B is a node, hop_A is the minimum number of hops to reach node A counting from some bootstrap node, N_A is a set of nodes which can be reached by node A in one hop, and $|N_A|$ is the number of nodes in N_A .)

Since, the distance between node B and A is at most 1 hop, $-1 \leq hop_B - hop_A \leq 1$; thus, we can guarantee that $0 \leq offset_A < 1$.

3.2. Process to Compute Hop Coordinates

In order to simplify communication, we assume that WSNs are symmetric (which means that if node

A can reach B , then B can reach A) and transitive. These assumptions are reasonable for this paper, since our technique is simulated above the 802.15.4 MAC layer [6] in NS-2 [4], which guarantees the communication to be symmetric (in NS-2, every message transmitted in the 802.15.4 MAC layer from node A to node B will be followed up by an *ACK* message from node B).

In a WSN which wants to compute a **hop coordinate**, we first appoint any one node as bootstrap node, after which the bootstrap node will generate a special message including a variable $hop=0$ with which it will flood the whole network, allowing all other nodes to count the hop distance from it. After a small constant TIMEOUT, all nodes can calculate their own **hop coordinate**. The detailed processes for both the bootstrap node and other nodes are shown in the following:

1. In bootstrap node:

Assigned bootstrap node initializes a message with $hop=0$ to flood the network; after that, the bootstrap node will drop any message which is originated by itself.

2. In any other nodes in WSN:

An algorithm like the following is run in any node A that is not the bootstrap node:

Algorithm in node $_A$

Input: message(hop_B) from node $B \in N_A$

Output: $hop_B, offset_A$

for each message(hop_B) from $B \in N_A$ and \sim TIMEOUT

if ($hop_B < hop_A$)

$hop_A = \min(hop_A, hop_B) + 1$;

forward (message(hop_A)) to MAC;

else

drop (message(hop_B));

endif

endifor

if ($|N_A| == 0$)

$offset_A = 0$;

else $offset_A = \frac{\sum_{B \in N_A} (hop_B - (hop_A - 1)) + 1}{2(|N_A| + 1)}$;

4. Simulation Results

4.1 Simulation Process

There are three steps in our simulation.

Step 1: Compute **hop coordinate** in NS-2

We implement the hop coordinates algorithm as a routing agent and the bootstrap node program as a protocol agent in NS-2 version 2.29 [4] with 802.15.4 MAC layer [6] and the CMU wireless extension [5]. The configurations for NS-2 are RF range = 15

meters, propagation = TwoRayGround, and antenna = Omni Antenna.

We use uniform placement— n nodes are placed on a grid with 100% of r randomized placement error. Here r is the width of a small square inside the grid. We constructed a total of 60 placements with $n = 36, 100, 225, 400, 625, 900, 1600, 2025,$ and 2500 , and with $r = 2, 4, 5, 8, 10,$ and 12 meters, respectively. The reason we use uniform placement with 100% error is that usually such placement includes both node holes and islands in one placement as shown in Fig.1.

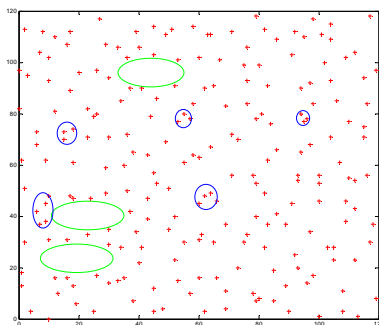


Fig.1 a typical placement for simulation
Constructed with $n=400, r=4$.
Green oval: holes, Blue ones: islands.

Then, we run the simulator on these placements to get a **hop coordinate** file for each placement.

Step 2: Compute predicted maps using localization algorithms

As described in the related work summary above, the MDS-MAP series of localization algorithms outperform other localization algorithm, so in our simulation we use MDS-MAP series algorithms to evaluate our technique. After we get **hop coordinate** files, we then use these files to compute shortest paths between all pairs of nodes using Dijkstra's algorithm, as described in [15] and [16]. Then, we apply MDS-MAP [15] and MDS-MAP(P) [16] to the above data in MatLab to compute the relative maps. Given four fixed anchor nodes, we transform these relative maps to absolute maps.

At the same time, we use the same placement to generate a connectivity map, and then input it to MDS-MAP and MDS-MAP(P) to generate the corresponding absolute maps, using the same four fixed anchor nodes used above.

Thus, for each placement, we now have four absolute maps created after this step.

Step 3: Evaluate inaccuracy of geographic routing on these absolute maps

There are several network terms, namely *non-optimal paths*, *packet drops*, and *routing*, that we initially identified as suitable to evaluate the impact of location inaccuracy on the performance of

geographic routing. In our experiment on the *routing* term, when we compared a route between source/destination pair nodes from one of four absolute maps with a route between the same source/destination pair nodes in the real map, we found that although all of the results from the four absolute maps are similar overall, there is nearly always at least one node that differs in the chosen routes. Therefore, we abandoned evaluation of this term. Likewise, we found the *packet drop* term largely relates to other network parameters such as buffer size and traffic, which are not key points we are considering here, and we also abandoned it. Thus, in the results we report we evaluate only the term *non-optimal paths*.

We use the following method to evaluate this term: A. Two nodes are randomly selected in a network, then one node is set as the source and the other as the destination. B. Let one message be sent from the source to the destination based on a geographic routing algorithm in each of the four absolute maps created in Step 2 above. At the same time, we let same geographic routing algorithm route the message propagation between the same pair of source/destination nodes in the real map. We judge that if the first node selected in the absolute map is not the same node selected in the real map, then this path is non-optimal; otherwise the path is optimal.

Most geographic routing algorithms, such as GPSR [11], GHT [12], and AODV [7], share a similar idea, that they should route the message to a node which is as near as possible to the destination. In order to simplify the problem, here we just select a simple but popular algorithm, AODV.

We conducted all experiments in NS-2, using the same configuration of the MAC layer as in Step 1 and with AODV selected as the routing algorithm. We did 1000 trials per absolute map in our experiments.

4.2 Simulation Results

We obtained results of the accuracy of geographic routing on the maps generated by MDS-MAP series algorithms, with and without hop coordinates, on network with $n = 36, 100, 225, 400, 625, 900, 1225, 1600, 2025,$ and 2500 nodes, and $r = 2, 4, 6, 8, 10,$ and 12 meters, respectively, as shown in Fig. 2. In the sub-figures, we use a diamond to represent the result of MDS-MAP running on connectivity only, a square to represent the result of MDS-MAP with hop coordinates, a triangle to represent the result of MDS-MAP(P) running on connectivity only, and an "x" to represent the result of MDS-MAP(P) with *hop coordinates*.

From Fig.2(a) we can see that when $r=2$, the accuracy in geographic routing with hop coordinates

is higher than it is in routing without them, except when $n \leq 100$. This is because when in a network of $r \leq 2$ and $n \leq 100$, at least 90% of all nodes in the 36-node network can reach with each other in one hop, and at least 40% of all nodes in 100-node network can reach with each other in one hop—this means that the diameter of a network with $r \leq 2$ meters and $n=36$ or 100 is just two or three hops. This situation is one in which our technique does not work well.

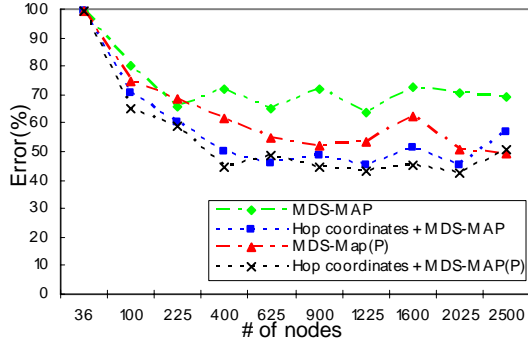


Fig.2.a comparison of the accuracy of routing ($r=2$)
Here: X-axis: # of nodes, Y-axis: the error of geographic routing in percent.

From Fig.2(b-d), we can see that when $4 \leq r < 10$ meters, the improvements of accuracy in geographic routing with hop coordinates are stable in the networks tested from a small scale (36 nodes) to the largest scale (2500 nodes).

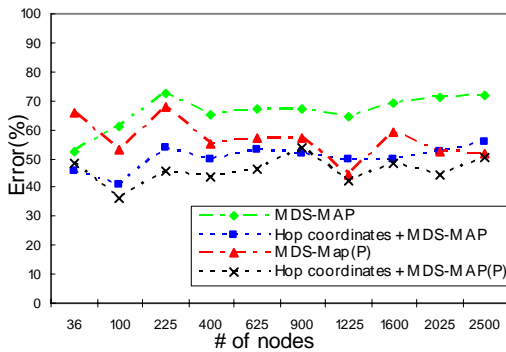


Fig.2.b comparison of the accuracy of routing ($r=4$)

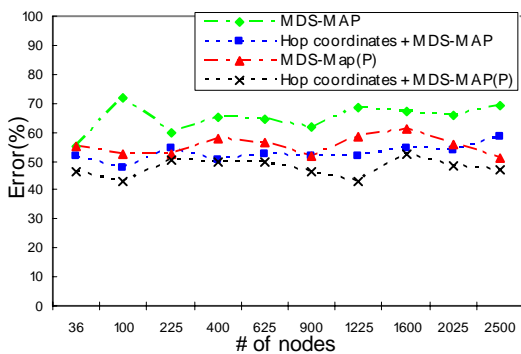


Fig.2.c comparison of the accuracy of routing ($r=6$)

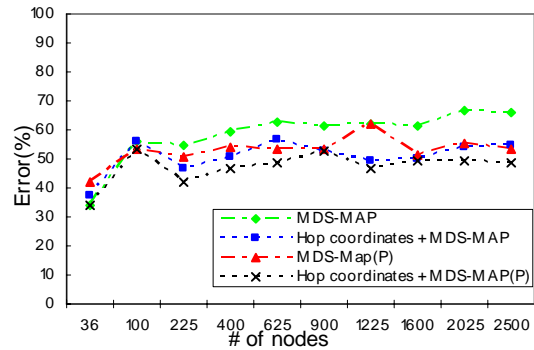


Fig.2.d comparison of the accuracy of routing ($r=8$)

Our technique doesn't require each node to connect to more than a certain number of neighbor nodes, but when $r \geq 10$, many nodes in the network have only one or two connections to other nodes. Figs.2(e) and (f) shows that in such a situation the hop coordinate technique has worse performance than when $r < 10$.

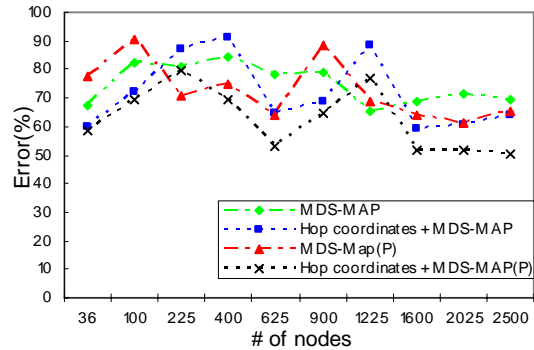


Fig.2.e comparison of the accuracy of routing ($r=10$)

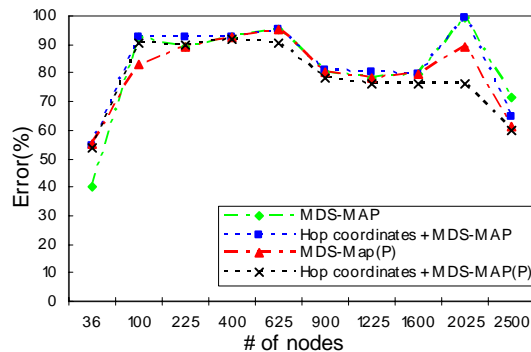


Fig.2.f comparison of the accuracy of routing ($r=12$)

Fig.2 comparison of the accuracy of geographic routing on the results of MDS-MAP series algorithms with or without the hop coordinates technique, on networks with $n = 36, 100, 225, 400, 625, 900, 1225, 1600, 2025,$ or 2500 nodes, and $r=2, 4, 6, 8, 10,$ or 12 meters, respectively

Notes: X-axis: # of nodes, Y-axis: the error of geographic routing in 100 percent.

