TrinaryMC: Monte Carlo Based Anchorless Relative Positioning for Indoor Positioning

Steven M. Hernandez and Eyuphan Bulut
Department of Computer Science, Virginia Commonwealth University
401 West Main St. Richmond, VA 23284, USA
{hernandezsm, ebulut}@vcu.edu

***Abstract*—Identifying positions of mobile devices within indoor environments allows for the development of advanced applications with context and environmental awareness. Classic localization methods require GPS; an expensive, high power consuming and inaccurate solution for indoor situations. Relative positioning allows nodes to recognize their location in relation to neighboring nodes to develop an internal mapping of their own position compared to those around them. This enables a quick deployment of a given system in new, unknown indoor environments without requiring prerequisite human mapping steps. In this paper, we develop a Monte Carlo Localization (MCL) based anchorless, relative positioning algorithm which simplifies the problem to considering three states of interaction between devices: approaching, retreating and invisible. Considering three states contributes to existing MCL methods which so far only consider binary states of visible or invisible. Through our anchorless approach, we show by simulations that TrinaryMC can provide more accurate positioning information than existing anchor based methods without relying on GPS, hence decreasing hardware costs and energy consumption from the use of GPS modules as well as reducing communication overhead compared to state-of-the-art MCL methods.**

I. Introduction

Indoor localization aims to identify people, robots or assets as they move through indoor settings where ubiquitous solutions such as GPS are unable to provide accurate results. Localization in these settings are utilized in applications for different purposes such as tracking objects and navigation. Primarily, indoor localization is achieved by obtaining GPS-based coordinates or by recognizing the closeness of specific static targets such as at a key entrance or exit to a building or commonly used stairwells which may be identified by unique signatures on a device's sensors such as unique radio spectrum fingerprint [1]–[3]. These systems not only require high setup costs in both materials and human involvement, but they also suffer from diminishing returns as models produce less accurate results as environments change over time.

Relative positioning on the other hand aims to provide similar features in dynamic environments. Instead of judging the closeness of a node to some static landmark, closeness to neighboring dynamic nodes within the network are considered. By this, no additional infrastructure is required for static-landmark recognition. Further, because a level of uncertainty is inherent to the expectations of relative positioning, the model should be generalized so as to provide useful results without requiring a time consuming manual human-setup stage.

In this paper, we study relative positioning of mobile nodes which communicate in an ad hoc and opportunistic fashion. We aim to locate the positions of nodes relative to each other while minimizing the required data communication between any pair of neighboring nodes. Many indoor localization and relative positioning solutions [4]–[6] use Received Signal Strength Indication (RSSI) levels to judge the physical distance of devices to one another, however this can result in poor estimations due to the high variance in RSSI values. Additionally, because the rate of change of RSSI is so rapid, sharing this constantly updating RSSI value to neighbors beyond one communication hop would cause high packet congestion; thus, it can overwhelm the network bandwidth. Instead, we look to remove this issue by replacing RSSI with three simple states: *approaching*, *retreating* and *invisible*. We develop a Monte Carlo Localization (MCL) algorithm which considers this trinary state information to perform predictions. Existing state-of-the-art MCL algorithms [7]–[10] only consider the binary state of *visible* and *invisible*. We evaluate the capabilities of our proposed solution through simulations to show that TrinaryMC can produce higher accuracy than existing MCL methods. Furthermore, because existing MCL algorithms require GPS enabled anchor nodes, our anchorless algorithm removes the need for high energy consuming GPS hardware while also minimizing the amount of communication overhead required.

The rest of the paper is structured as follows. We discuss the related work in Section II. In Section III, we describe the details of the proposed approach. In Section IV, we provide our simulation results. Finally, we provide concluding remarks and outline the future work in Section V.

II. Related Works

Early work in indoor localization employed Radio Frequency Identification (RFID), successfully applying the technology to fields such as manufacturing and healthcare [11], [12]. However RFID localization requires the use of static readers placed throughout a building to supply adequate coverage in addition to specialized equipment to tag items which are to be localized. Bluetooth beaconing takes a similar approach

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 1744624. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

978-1-7281-3893-0/20/$31.00 ©2020 IEEE

by replacing RFID with Bluetooth Low Energy (BLE) [6] [13]. The primary advantage of such a system is the ubiquity of BLE enabled smart devices specifically in applications requiring the tracking of human subjects. Still, static hardware must be placed throughout a building to provide adequate coverage.

Relative positioning has been the interest of more recent work in identifying the location of devices in indoor settings. Studies such as DiscoveryTree [14] create a multi-hop mesh tree network which allows devices to simply recognize whether a given node is within the network and if so, which other nodes are in range. Bellrock [15] utilizes the mobile devices as Bluetooth beacons when they are stationary and stops beaconing when the devices move to prevent tracking of individuals' movements for privacy preservation. Many other works [4], [5] take the approach of determining the distance of pairs of nodes through RSSI readings, often employing Multi-dimensional Scaling (MDS) approaches. However, sharing constantly changing RSSI information between neighbors could very easily overwhelm the bandwidth of the network. Furthermore, as argued in [16], without extensive and time-consuming testing, raw RSSI values are suggested to be unacceptable for use in mapping to distances between devices. The authors do suggest that by testing devices, parameters can be set to achieve slightly higher results, however as shown in [6], [17], radio hardware between different manufacturers and even those produced in the same product line can produce very different RSSI values. Thus, because testing cannot be completed in any large network between each pair of devices, RSSI cannot be an adequate solution. Some works [18] consider not the distance but instead the relative velocity between nodes while others [19] remove the use of unreliable RSSI by only considering binary contacts between nodes.

One method popularized for use in decentralized localization is Monte Carlo Localization [20]. These techniques simulate internally multiple possible situations a given mobile node may be in subject to some constraints such as location of neighbor node and transmission range of a given neighbor node to choose a most likely final predicted location. Existing MCL methods expect the presence of anchor nodes; nodes with access to their exact location through GPS or some other method. For example, solutions in [10], [21] rely exclusively on the presence of anchor nodes to guide predictions. Other works such as [7], [8], [22] relax this constraint by using non-anchor nodes in addition to anchor nodes when making predictions. However, none of the existing works specifically focus on the cases when no anchors are present. Our aim is to provide a method which does not rely on anchor nodes in order to have low hardware cost per node and less reliance on a small group of nodes within the network. Because our method itself is an MCL algorithm, we now introduce the details of the four state-of-the-art MCL algorithms which we will use for comparison in Section IV.

A. Standard Monte Carlo Localization (St-MCL)

Each MCL method employs a very similar set of procedures for predicting positioning. The idea is to collect a group of samples in each time instance which can then be used in aggregate to determine the most probable location for the given node. All MCL algorithms; including our own, execute the following general procedure:

- *Initialization Step* - occurs only at startup or when all samples are filtered out. Creates an initial set of random samples subject to some conditions.
- *Sampling Step (Move Samples)* - using samples in previous time instance, new samples are created by moving previous samples randomly within a radius of v_{max} (i.e., maximum velocity) around the previous sample.
- *Filtering Step (Resample)* - after predicting new locations, any sample not adhering to a given set of constraints are filtered out as invalid samples. If the number of samples left after filtering is above some threshold, then random samples are filtered out to prevent sample sets from exploding in quantity over multiple rounds.

This simple method as described in [21] is the basis for our first comparison implementation. This algorithm relies entirely on accurate anchor nodes to achieve reliable results.

B. Orbit (Orbit-MCL)

One of the areas where common improvements are made to MCL methods is in minimizing the possible area with which samples are taken from. One such work is [8] where the authors consider graph theory when developing a new MCL method called Orbit. One of the key features they recognize is that when considering negative information (node A cannot see B), there are cases when multiple disjoint regions may occur in the sampling stage. In this work, they determine that ignoring regions with fewer than one-third the number of samples as the largest region produces much higher accuracy.

C. Low Communication Cost Monte Carlo (LCC-MCL)

Each of the previous methods relies exclusively on anchor nodes to inform their predictions, however the LCC-MCL method [7] considers not only anchor nodes but also normal, non-anchor nodes. Non-anchor nodes share their own predicted location with selected neighbors which then use the location, along with appropriate weights, to predict their own location. The issue with this is that sharing predicted locations of each and every node requires high numbers of packets to update the state between neighbors. The authors mitigate this issue by only sharing locations with neighbors which are believed to be close by. Determining whether nodes are close by one another is a matter of finding the intersection of each node's set of neighbors. If the number of intersecting neighbors is above a threshold, the nodes are considered close by, in which case, communication occurs.

III. TrinaryMC Algorithm

Our TrinaryMC method improves the existing state-of-the-art Monte Carlo Localization methods by entirely removing reliance on GPS enabled anchor nodes to produce relative positioning. In addition, our method is able to infer additional information about the state of neighboring nodes beyond

what is common in all existing methods while decreasing communication overhead compared to existing methods. Our method relies on first recognizing the following trinary states occurring between pairs of neighboring nodes: *approaching* (1), *retreating* (2) and *invisible* (0), where each state value is identified in our system as the integer shown in parenthesis. Each node transmits a beacon packet at given intervals to announce their presence. Neighboring nodes recognize this packet and record the presence of this neighbor along with the perceived state: *approaching* or *retreating*[1]. Each node can further propagate the list of its one-hop neighbors to its neighbors and let the nodes recognize the network structure [23] up to some number k-hops away. Our algorithm is applicable for any value of $k > 0$; however, we find that smaller values such as $k = 2$ result in adequate accuracy while keeping the messaging overhead low. To collect the current state between all neighbors, we create a local state matrix, M_t, and denote the current state between two nodes i and j at time t by $M_t^{(i,j)}$.

A. Differences with existing MCL algorithms

1) Initialization Step: In existing MCL algorithms, each sample contains only one (x, y) coordinate pair prediction for a given source node s, however in our method, a single sample must include an (x, y) coordinate pair p for each neighbor of s. To accomplish this, the predicted coordinate pair for s is set to $p^{(s)} = (0, 0)$. Then, for each neighbor n, we determine a random $p^{(n)}$ satisfying our state matrix $M_t^{(s)}$. One important factor to recognize while collecting initial samples is that even though we concern ourselves with trinary states, in the initialization step, the only states that matter are *invisible* and *visible* because no information exists from previous time instances to distinguish *approaching* from *retreating*. After the initialization step is completed, we repeat the sampling and filtering step for each future time instance.

2) Sampling Step (Move Samples): In the sampling step, the goal is to take the set of samples S from $t - 1$ and move each neighbor n so that M_t continues to be satisfied at time t. Creating new samples based on samples from previous time instances is common to other MCL methods, however, because our goal is unique in recognizing trinary states, the method which we accomplish this is novel to MCL. Common to existing MCL algorithms, in this step, each neighbor is moved at most v_{max} distance from its previous location. Four cases exist: first, the case when a neighbor keeps the same state from $t - 1$ to t. The second case is when a neighbor is visible (retreating) at $t - 1$ and then is invisible at t. In this case, we must ensure that our sample point exits the range of the given paired node. The third case is when a neighbor is invisible at $t - 1$, but visible at t (approaching). In this case, we do not have any predictions of where the node was at $t-1$, so we do not consider any previous information. Instead, we simply randomly select a point which satisfies all conditions in M_t. Finally, we handle the case of either *approaching* to *retreating* or *retreating* to *approaching*.

[1]Through our experiments with actual devices, we find that a smartphone can identify these states with $> 95\%$ accuracy by using the change in RSSI from a given neighbor, even between smartphones of different manufacturers.

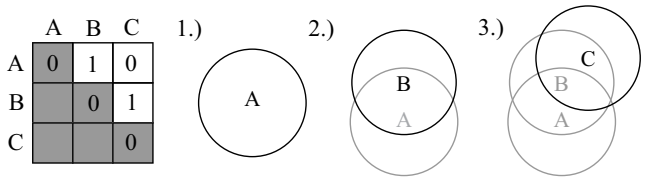

Fig. 1. Process taken to generate a single sample given a state matrix $M_t^{(A)}$ shown on the left containing 3 k-hop (where k=2) neighbors A, B and C. The process takes 3 steps, one for each neighbor. In the first step, the current source node A is placed at position $(0, 0)$. The second step places neighbor B within the radius of A because $M_t^{(A,B)} = 1$ indicating A can see B. The third step places C based on the fact that B sees C, but A does not see C.

3) Final Predictions: The final location of a node relative to its neighbors is predicted from all samples in S. To this end, we simply take the average distance of each pair of nodes $(i, j) \in K$ where K is a list of neighbors and $i \neq j$:

$$d_{pred}^{(i,j)} = \Sigma_{s \in S} ||s_i - s_j|| \quad (1)$$

where $||s_i - s_j||$ is the Euclidean distance between sample points s_i and s_j.

B. Generating Samples

As the general steps taken by the entire algorithm have been discussed, we now provide the details of our method for generating samples. Fig. 1 provides an example case which we will use to illustrate a given iteration of generating samples. We begin by assuming the role of a node A with three neighbors (K), including itself. Given K, we assume at A we can collect M_t. In the figure, we see the values of M_t on the left. With M_t, we begin the process of creating a sample by randomly selecting positions for each neighbor in K. In the first step, we place A by default at $(0, 0)$, the local center. In the second step, we place B randomly. Because $M_t^{(A,B)} = 1$, the placement of B must be within the radius of A. In the illustration, this is successful, but in the event of a failure (at this step or any further steps in this process), we simply retry from the beginning of this process. Now that B has been placed, we move to the third step by placing C following the conditions specified in M_t. We can see, because $M_t^{(A,C)} = 0$, C is a 2-hop neighbor of A, thus we successfully complete this step with C in range of B. After all conditions in M_t are considered, we have created a single sample for S.

In subsequent steps of the algorithm, we take S from time t as a base to further create S' for time $t+1$ which adheres to the constraints of M_{t+1}. To accomplish this, we simply loop through each neighbor (n) in K_{t+1}. If the neighbor was also in K_t, then we use the coordinate value (x, y) from S to produce n in S'. The new coordinate must be within a radius of v_{max} of (x, y) in addition to adhering to the trinary conditions of the pair between t and $t + 1$. We can see this case in Fig. 2. Assuming A and B have been placed in S', the shaded area around C represents the maximum distance C can travel in

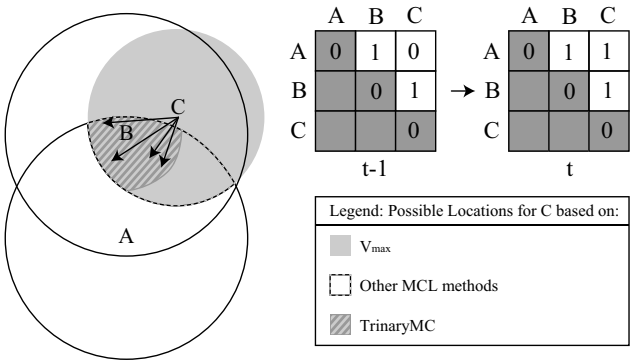

Fig. 2. Possible movement for neighbor C based on v_{max} is shown in gray, but not all of these locations are possible given the conditions in M_{t+1}. Any point in the area with the dashed perimeter (intersection of circles A, B and the v_{max} area) is valid for C's next location if only the contact matrix is considered as in the case of other MCL methods. Our TrinaryMC method further reduces the valid areas where C can move thanks to the consideration of three states; thus, it produces more accurate samples.

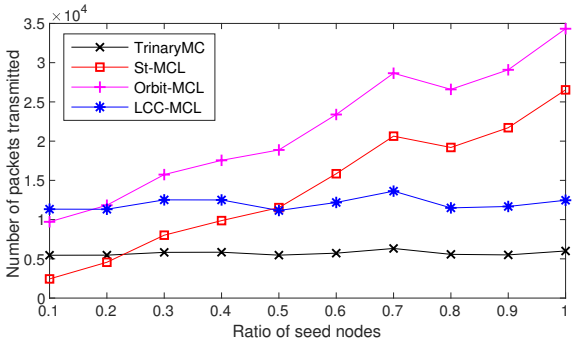

Fig. 3. As the ratio of anchor seed nodes increases, the number of packets communicated through the network also increases except for our TrinaryMC method as well as the LCC-MCL method.

this time instance. Because C must transition to *approaching* A after being *invisible* at time t, the region in which valid samples can be taken shrinks significantly.

C. Alternative Algorithms

Our *trinary* algorithm can be simplified in a couple of ways. We use these simplified algorithms to further evaluate our trinary approach later on. First, we consider a simple case where we assume no memory exists between $t-1$ and t for each node. In this case, we have no reason to move samples or resample. Thus, this first simplified algorithm only runs the initialization step at each time instance no matter if $|S| > 0$ or not in $t-1$. We call this the *binary-no-memory* method, binary because without memory of $t-1$, neither *approaching* nor *retreating* can be recognized.

A middle ground algorithm between this *binary-no-memory* algorithm and our *trinary* algorithm is running the *trinary* algorithm using only the binary states *visible* and *invisible*. For this, we continue to recognize the movement of samples between time instances; thus, we must retain the sampling step and the filtering step. We evaluate this second simplified algorithm in Section IV as the *binary* method. In the following section we show through simulations that over time, the *binary* method performs better than the *binary-no-memory* method and the *trinary* method performs better than the *binary* method as would be expected.

IV. Simulations

In this section, we compare our method to state-of-the-art MCL techniques, namely the Standard Monte Carlo Localization algorithm (St-MCL) [21], Orbit (Orbit-MCL) [8] and finally the Low Communication Cost MCL (LCC-MCL) [7]. In addition to comparing our trinary method to existing MCL methods, we also compare it to our binary-no-memory (bin-no-mem) and the binary methods as described in Section III-C. Our simulation uses a $500m \times 500m$ area where 50 mobile nodes move using the random waypoint mobility model with a velocity randomly assigned from $[1-10]$ m/s.

First, we consider the communication required for each method. The primary gain in this aspect for our TrinaryMC method compared to existing methods is that our method needs to share message with first and second hop neighbors only when a trinary state change occurs between a pair of devices. In St-MCL, updates are shared between each k-hop neighbor whenever an anchor node receives a new GPS location. Because we simulate time discretely, all anchor nodes move every single time instance. The Orbit-MCL and LCC-MCL methods also require communication for each anchor position change but also require updates from non-anchor nodes to provide additional accuracy in their methods. The LCC-MCL method keeps in mind this additional communication and provides a method to lower this cost. Instead of sharing GPS position updates from all anchor and non-anchor nodes to all neighboring nodes, a closeness metric is obtained to determine whether a given neighbor is close enough to gain insight from the new GPS position. As the ratio of anchor seed nodes increases, the amount of communication required also increases for all methods except for our TrinaryMC and LCC-MCL as can be seen in Fig. 3. For TrinaryMC, this is because the model does not consider anchors whatsoever. For LCC-MCL this occurs as a result of the use of the closeness metric to determine which nodes to communicate with instead of simply whether the node is an anchor or not. Closeness does not change as more nodes become anchor seeds.

Now, we consider the communication radius (r) of each node. As r increases, more neighbors on average are seen within any hop range of the nodes, as shown in Fig. 4. Note that, when r is smaller than 30m, the number of one-hop neighbors on average is smaller than 1. In these cases, as we cannot make predictions without any data from neighbors, we ignore them in our analysis.

We now consider how we evaluate the error for given sets of nodes after running each method. For this, we simply take the difference for each sample prediction compared to the actual distance of each given node pair. Fig. 5 shows the error obtained in our three methods, binary no-memory, binary and

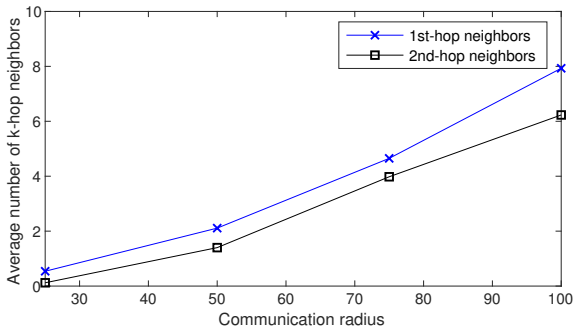

Fig. 4. Average number of k-hop neighbors seen given some communication radius (r).

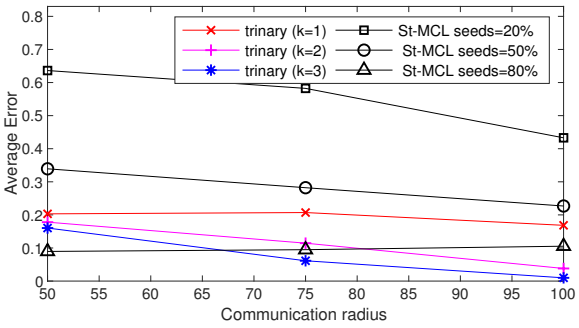

Fig. 6. TrinaryMC method comparing to the Standard Monte Carlo Localization (St-MCL) methods for different number of anchor seeds in the network.

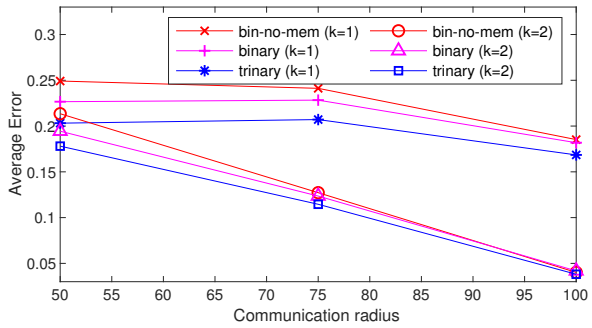

Fig. 5. Comparison of our methods as communication radius changes (thus more neighbors are seen per node) and as k increases (also causing more neighbors to be seen).

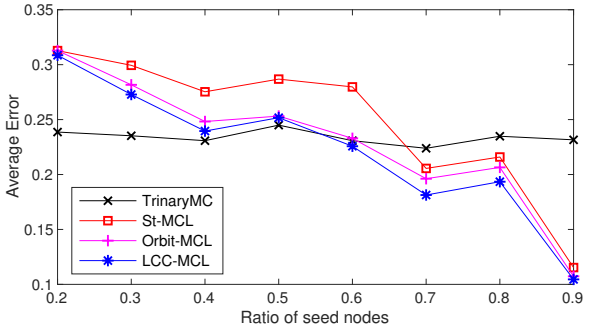

Fig. 7. Comparison of TrinaryMC to state-of-the-art MCL methods when different percentage of the nodes in the network are marked as anchor nodes and communication radius, $r = 50$, and $k = 2$.

trinary. We take the calculated average error per node after 100 time steps divided by the communication radius (r) to produce a percent error relative to r. We can see that as radius increases, error decreases, which can be attributed to the fact that more neighbors are seen with higher r. Further, we can see when $k = 2$, we achieve better accuracy than we do with $k = 1$. We can attribute this increase in accuracy again to more neighbors, which help create fewer possible locations when generating samples. We also observe that our trinary method provides lower error than the binary method, which itself produces lower error than the binary no-memory method.

Existing MCL algorithms require anchor nodes or *seed* nodes which can sense their exact location at any given time through a method such as GPS. Because our method is anchorless, TrinaryMC is not affected by an increase of seeds. We explore how changing the percentage of seed nodes within the network affects St-MCL and how TrinaryMC compares to St-MCL as a result in Fig. 6. We can see when only 20% of nodes are labeled as seeds, accuracy decreases significantly, ranging from 40% and 70% of r. However, with 50% of nodes as seeds, we begin to reach the same average error as in our trinary case. Another observation we can make from this figure is that as communication radius for nodes increases, average error decreases. This is explained by the fact that a larger communication radius reveals more neighbors which then results in a more restricted area when predicting samples. Next, with $r = 50$ and $k = 2$, Fig. 7 compares our algorithm with additional state-of-the-art methods. We again see the trend where a lower ratio of seeds (below 50%) produces worst accuracy results than TrinaryMC for each of the existing state-of-the-art algorithms. We also notice that both Orbit-MCL and LCC-MCL perform better than St-MCL when the ratio is between 0.3 and 0.7, but perform similarly to one another. Having a high ratio of seed nodes in the network however is not reasonable because of considerations such as hardware cost as well as the issue of battery consumption from modules such as GPS. Thus, even though the solution may provide better results with greater number of seeds, the solution is highly impractical and costly.

We finally compare the RSSI based distance estimation with our method. Many existing works use the Log-distance path model to explain how RSSI is affected by distance in given path-loss exponent (n) [24]. The value for n implies the amount of noise and environmental effect on the RSSI. Surveying existing works [24], [25], we see common values for n between 2 and 4. In Fig. 8, we show how different values for n affect the average error from the low end of $(n, n+0.5)$ up to $(n, n+1.0)$. We can see the average error with RSSI based distance estimation is equal when $n = 2$ and better on average as n increases than the binary-no-memory method. However, we can see that our trinary method performs better on the average until n reaches 2.8. Still, in the worst case, we

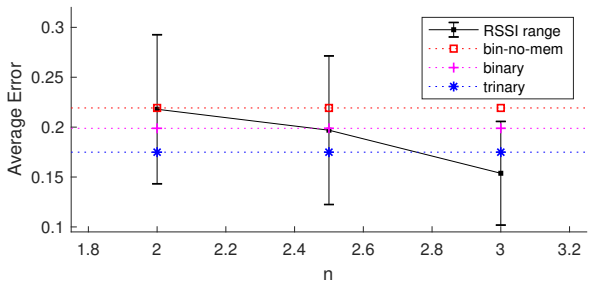

Fig. 8. Results of simulation of RSSI values with different path-loss exponent and variance, and their comparison with our methods.

can see that RSSI performs worse than the binary method up to $n = 3$. This suggests that our method can perform as good as the RSSI method or better in very uncertain environments such as highly congested areas with more noise or indoor environments with high signal reflections.

V. Conclusion

In this work, we present an anchorless relative positioning method which utilizes multi-hop neighbor state information using the trinary states: *approaching*, *retreating* and *invisible*. We developed a Monte Carlo Localization algorithm using these states to produce relative positioning results better than existing Monte Carlo Localization methods which rely on anchor nodes present within the network. Because of our improvement of removing all reliance on anchor nodes, our method can decrease energy consumption from the use of GPS radio hardware, decrease hardware cost by removing the need for GPS modules whatsoever on all device, and decrease the reliance on the limited individual anchor nodes in a system. Additionally, we show that our method results in less communication overhead compared to state-of-the-art algorithms. Our method benefits as more devices join the system as each not only contributes information about themselves, but also their own k-hop neighbors, thus giving each node a further look into the network allowing for more accurate sampling.

References

[1] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, "No need to war-drive: Unsupervised indoor localization," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 2012, pp. 197–210.

[2] M. Alzantot and M. Youssef, "Crowdinside: automatic construction of indoor floorplans," in *Proc. of the 20th International Conf. on Advances in Geographic Information Systems*. ACM, 2012, pp. 99–108.

[3] F. Yucel and E. Bulut, "Clustered crowd gps for privacy valuing active localization," *IEEE Access*, vol. 6, pp. 23 213–23 221, 2018.

[4] R. T. Rajan, G. Leus, and A.-J. van der Veen, "Relative kinematics of an anchorless network," *Signal Processing*, vol. 157, pp. 266–279, 2019.

[5] P. Abouzar, D. G. Michelson, and M. Hamdi, "Rssi-based distributed self-localization for wireless sensor networks used in precision agriculture," *IEEE Transactions on Wireless Communications*, vol. 15, no. 10, pp. 6638–6650, Oct 2016.

[6] D. Chen, K. G. Shin, Y. Jiang, and K.-H. Kim, "Locating and tracking ble beacons with smartphones," in *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies*. ACM, 2017, pp. 263–275.

[7] A. M. A. Abu znaid, M. Y. I. Idris, A. W. A. Wahab, L. K. Qabajeh, and O. A. Mahdi, "Low communication cost (lcc) scheme for localizing mobile wireless sensor networks," *Wireless Networks*, vol. 23, no. 3, pp. 737–747, Apr 2017. [Online]. Available: https://doi.org/10.1007/s11276-015-1187-6

[8] S. MacLean and S. Datta, "Reducing the positional error of connectivity-based positioning algorithms through cooperation between neighbors," *IEEE Transactions on Mobile Computing*, vol. 13, no. 8, pp. 1868–1882, 2013.

[9] J. Radak, L. Baulig, D. Bijak, C. Schowalter, and H. Frey, "Moving towards wireless sensors using rssi measurements and particle filtering," in *Proceedings of the 14th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, ser. PE-WASUN '17. New York, NY, USA: ACM, 2017, pp. 33–40. [Online]. Available: http://doi.acm.org.proxy.library.vcu.edu/10.1145/3134829.3134839

[10] H. Chen, F. Gao, M. Martins, P. Huang, and J. Liang, "Accurate and efficient node localization for mobile sensor networks," *Mobile Networks and Applications*, vol. 18, no. 1, pp. 141–147, 2013.

[11] T. Sanpechuda and L. Kovavisaruch, "A review of rfid localization: Applications and techniques," in *2008 5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, vol. 2, May 2008, pp. 769–772.

[12] J. Zhou and J. Shi, "Rfid localization algorithms and applications—a review," *Journal of Intelligent Manufacturing*, vol. 20, no. 6, p. 695, Aug 2008. [Online]. Available: https://doi.org/10.1007/s10845-008-0158-5

[13] S. A. Cheraghi, V. Namboodiri, and K. Sinha, "Ibeaconmap: Automated indoor space representation for beacon-based wayfinding," *arXiv preprint arXiv:1802.05735*, 2018.

[14] O. Chabloz, D. D. S. ndrade, A. Upegui, H. F. Satizbal, and A. Perez-Uribe, "Discoverytree: Relative localization based on multi-hop ble beacons," in *2017 Global Internet of Things Summit (GIoTS)*, June 2017, pp. 1–6.

[15] A. Zidek, S. Tailor, and R. Harle, "Bellrock: Anonymous proximity beacons from personal devices," in *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2018, pp. 1–10.

[16] K. Heurtefeux and F. Valois, "Is rssi a good choice for localization in wireless sensor network?" in *2012 IEEE 26th International Conference on Advanced Information Networking and Applications*, March 2012, pp. 732–739.

[17] G. Lui, T. Gallagher, B. Li, A. G. Dempster, and C. Rizos, "Differences in rssi readings made by different wi-fi chipsets: A limitation of wlan localization," in *2011 International Conference on Localization and GNSS (ICL-GNSS)*, June 2011, pp. 53–57.

[18] S. Kumar, R. Kumar, and K. Rajawat, "Cooperative localization of mobile networks via velocity-assisted multidimensional scaling," *IEEE Transactions on Signal Processing*, vol. 64, no. 7, pp. 1744–1758, 2016.

[19] Z. Wang, E. Bulut, and B. K. Szymanski, "Distributed target tracking with imperfect binary sensor networks," in *IEEE Global Telecommunications Conference (GLOBECOM)*. IEEE, 2008, pp. 1–5.

[20] A. M. A. A. Znaid, M. Y. I. Idris, A. W. A. Wahab, L. K. Qabajeh, and O. A. Mahdi, "Sequential monte carlo localization methods in mobile wireless sensor networks: A review," *Journal of Sensors*, vol. 2017, pp. 1–19, 2017. [Online]. Available: https://doi.org/10.1155/2017/1430145

[21] A. Alaybeyoglu, "An efficient monte carlo-based localization algorithm for mobile wireless sensor networks," *Arabian Journal for Science and Engineering*, vol. 40, no. 5, pp. 1375–1384, 2015.

[22] A. M. Khedr, "New localization technique for mobile wireless sensor networks using sectorized antenna," *International Journal of Communications, Network and System Sciences*, vol. 8, no. 09, p. 329, 2015.

[23] E. Bulut, Z. Wang, and B. K. Szymanski, "The effect of neighbor graph connectivity on coverage redundancy in wireless sensor networks," in *2010 IEEE International Conference on Communications*. IEEE, 2010, pp. 1–5.

[24] S. Shue and J. M. Conrad, "Procedurally generated environments for simulating rssi-localization applications," in *Proceedings of the 20th Communications & Networking Symposium*. Society for Computer Simulation International, 2017, p. 7.

[25] H. Karvonen, K. Mikhaylov, M. Hmlinen, J. Iinatti, and C. Pomalaza-Rez, "Interference of wireless technologies on ble based wbans in hospital scenarios," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Oct 2017, pp. 1–6.