An analysis of recursive cooperative indoor localization approaches

Heiko Will, Jakob Pfender, Stephan Adler, Thomas Hillebrandt, Marcel Kyas
Institute of Computer Science
Freie Universität Berlin
Takustr. 9, 14195 Berlin, Germany
Email: {heiko.will,jakob.pfender,stephan.adler,thomas.hillebrandt,marcel.kyas}@fu-berlin.de

***Abstract*—We present a conclusive comparison of two different cooperative localization schemes, namely the successive refinement and network multilateration approaches. Both strategies use ranging information to localize nodes in the network, differing in the form this ranging information takes. Successive refinement has nodes estimate their positions by ranging to their neighbors, who in turn estimate their positions in the same way. Network multilateration ranges by finding the shortest multihop path to the network anchors and using the length of these paths as ranging information.**

We test both these approaches in a variety of scenarios involving mobility using a JiST/SWANS based simulator.

We show that network multilateration, even though it introduces an additional error source, can achieve better results than successive refinement in most scenarios involving mobility.

***Index Terms*—Indoor Localization, Cooperative Localization, Localization Algorithm, Wireless Sensor Networks, Simulation**

I. Introduction

While localization in outdoor environments is a solved task, the problem of a reliable ad-hoc indoor positioning system still poses great challenges.

We put our focus on radio based localization systems which use range based lateration algorithms. Within this domain one of the most important issue is to get an accurate value for the distance between a network member with a known position called anchor and a network member with unknown position called node. The accuracy of the measured distance between anchor and node has a high impact on the accuracy of the position which is a result of multiple measured distances used by an lateration algorithm. Besides the problem of the distance measurement itself the problem of link quality occurs within real world radio networks. To find the position of a node we need to know the distance between the node and at least three anchor nodes. To measure this distance the node needs a stable radio link to a sufficient amount of anchors. In a real world scenario we will see a lot of cases where this requirement can not be fulfilled for all nodes in the network.

The solution to this problem is to use cooperative localization, which means that the node which does not have direct access to a sufficient amount of anchors can use other neighbour nodes as anchors using their measured position. This is achieved by recursion — the network multilateration approach has nodes recursively discovering paths to anchors using other nodes as intermediaries and then using the length of these paths as ranges, whereas the successive refinement approach has nodes with direct contact to anchors estimate their positions and then acting as new anchors to other nodes, in turn allowing them to estimate their position. The latter, however, entails a strong possibility for errors in scenarios involving mobility, as nodes that move out of the reach of all anchors still use the recursively estimated — and thus error-prone — positions of their neighbors, making their own estimates even less accurate. This so-called ”clustering” error can grow very large very fast, rendering all position estimates within the clusters useless. We show that network multilateration is not affected by this problem.

II. Related Work

The problem of cooperative localization is well known in the domain of wireless sensor networks and a lot of different algorithms and approaches have been discussed in the last decade. A very good overview of the work of the last years is presented by Patwari et al. [1]. The authors distinguish between *centralized* algorithms and *distributed* algorithms to solve the problem of localization within a sensor network. As we assume in our case that we to not have a central entity which shall solve the task and want to reduce the volume of communication within the network we will focus on the distributed algorithms.

The distributed algorithms are also split into two different approaches which Patwari et al. call *network multilateration* and *successive refinement* [1].

The network multilateration approach was introduced by Niculescu [2] and also independently by Savvides et al. [3]. In both works the shortest paths between three or more anchor nodes and a node is searched in a network and afterwards used as distance value for the multilateration algorithm. We use the same way of measuring distances in our simulation, but apply several other localization algorithms in addition to multilateration.

With the successive refinement algorithm a more intuitive way of collaborative localization was presented independently by several authors [4]–[6]. The basic idea in this approach is to give each node a position estimate in a certain way, e.g. by using network multilateration and to improve these results afterwards by recursively gathering better position estimates. Therefore nodes which have a 1-hop connection to

anchors localize themselves and distribute to the network when their position estimate has converged to a certain accuracy. Afterwards these nodes can also be used as anchors and so forth. Theoretically this approach seems to result in better position estimates network wide, a claim which we wanted to examine in our work.

In this paper we want to compare the performance of different cooperative localization approaches. Savvides et al. [7] provide an overview of the parameters we need to take into account when analyzing the accuracy of cooperative localization, such as network density, path length and localization error itself. Whitehouse and Culler [8] compared different cooperative localization systems by using real world localization data to check whether the theoretically developed metrics and bounds of former work can be used in real world scenarios.

In our work we compare the network multilateration and the successive refinement approaches using the JiST/SWANS simulation environment.

III. Simulation

A. JiST/SWANS

We implemented our approach using the JiST/SWANS[1] simulation environment, which is described in detail by Barr [9], [10]. We used a modified set of the components provided by the SWANS framework, including radios that simulate interference, packet collisions and packet loss.

B. Algorithms

Using our simulation environment, we implemented both the network multilateration and the successive refinement approaches in order to compare them using the same parameters.

In both approaches, anchors with a priori location information broadcast their position to all other nodes. The other nodes gather these broadcasts in order to fill their routing tables. In the network multilateration approach, the routing table tracks the current estimated distances to all anchors as well as the position and anchor distances of all neighbors (anchors and non-anchors). Entries are updated whenever a broadcast is received. The routing table keeps track of the shortest paths to all anchors (see Algorithm 1).

The successive refinement approach similarly uses routing tables — it stores the positions of nodes it can hear together with their hopcount (see Algorithm 2).

In order to locate themselves, nodes first retrieve the best entries from their routing table. In the network multilateration approach, this means selecting the shortest path to all reachable anchors as previously stored in the routing table, whereas in the successive refinement approach it means selecting those neighbors with the lowest hopcounts and using their estimated positions. In any case, the node now has the positions of at least three nodes as well as their distances and hands these over to the lateration algorithm of choice. It is important to note here that the choice of method for lateration has nothing to do with the choice between successive refinement and network multilateration — this choice is confined solely to the question of how the localization information is obtained. The lateration algorithm can be chosen independently. However, as we will show later, the quality of the results depends on the combination of localization method and lateration algorithm.

[1] **J**ava **i**n **S**imulation **T**ime / **S**calable **W**ireless **A**d-hoc **N**etwork **S**imulator

Algorithm 1 Network multilateration — routing table update

```
if neighbor.isAnchor() then
    range ← estimated distance
    routingTable.updateAnchorEntry(neighbor)
end if
if neighbor already in routing table then
    neighborEntry.setLocation(loc)
    neighborEntry.setHopCount(hopCount)
    neighborEntry
        .updateAnchorDistances(anchorDistances)
else
    routingTable.newNeighborEntry(
        loc, hopCount, anchorDistances)
end if
```

Algorithm 2 Successive refinement — routing table update

```
if neighbor already in routing table then
    neighborEntry.setLocation(loc)
    neighborEntry.setHopCount(hopCount)
else
    routingTable.newNeighborEntry(loc, hopCount)
end if
```

As has been shown, the two approaches differ solely in the parameters they hand over to the lateration algorithm — one uses the recursively determined locations of its neighbors, the other approximated distances to the anchors. All other aspects are identical, allowing us to conclusively compare the different localization strategies.

C. Parameters

We tested our implementation using a $150 \times 150m^2$ field with three anchors arranged in a triangle in the upper left corner. The mobile nodes were initially placed in the center of the anchor triangle and then started moving according to a "random waypoint" mobility model, meaning that each individual node chooses a random point on the field, moves towards it for a given time with a random speed between 0.1 and $2m/s$, pauses for a given time and repeats the process. This will distribute the mobile nodes evenly across the field during the course of the simulation run. Variable parameters were the number of mobile nodes ($\{25,100,200,300\}$), the number of anchors ($\{3,4,5\}$, with additional anchors after the first three being placed randomly in the field), update frequency (location is updated every $\{1,2,3,4\}$ seconds) and the lateration algorithm used (Trilateration, LLS, NLLS, AML).

Because our approach is a cooperative one, we can improve our accuracy by increasing the number of mobile nodes. Our results show that the lower bound on mobile nodes for a

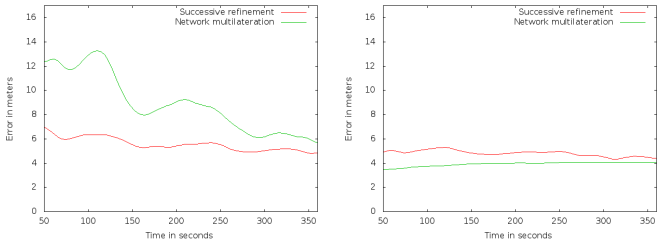


Fig. 1. Comparison of the strategies in low and high node density scenarios. Left: 25 mobile nodes, right: 100 mobile nodes

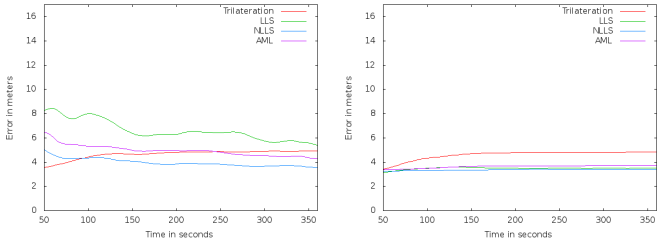


Fig. 2. Comparison of different combinations of localization strategy and lateration algorithm. Left: Successive refinement, right: Network multilateration

$150 \times 150m^2$ field lies around 50 nodes. As can be seen in Fig. 1, at a density level of 25 nodes, network multilateration is much worse than successive refinement, with successive refinement achieving only marginally worse results than at higher densities. Any number of nodes above 50 achieves acceptable results for both strategies, with accuracy still improving noticeably when going from 50 up to 300 nodes. However, above a density of 50 nodes, network multilateration quickly becomes better than successive refinement, rapidly overtaking the other approach at a density of around 100 nodes.

In most traditional localization approaches, accuracy can easily be boosted by increasing the number of anchors in the field. In comparison, our experiments show that our approach is not significantly improved by increasing this parameter. This is due to the cooperative nature of our algorithm — no node needs to be directly within range of three anchors, it just needs to have access to multihop paths to three anchors. Another obvious method to increase accuracy is to increase the frequency of location updates. In traditional approaches, where nodes rely on inaccurate position estimates of their neighbours, frequent updates are essential to keep these estimates realistic. In our approach, where anchor positions are fixed and only the length of the multihop paths between nodes and anchors varies, we can afford data being a little staler.

Finally, position accuracy is obviously strongly influenced by the lateration algorithm used to process the results obtained by localization. For our simulation, we tested the two localization strategies in conjunction with four well-known lateration algorithms: Classic trilateration, Linear Least Squares (LLS), Non-Linear Least Squares (NLLS) and Adaptive Multilateration (AML). All except the first of these are multilateration algorithms, meaning that they can utilize additional anchors other than the three needed for trilateration in order to further refine their position estimate. The goal of this paper is *not* to compare these lateration algorithms, but to analyze the results obtained from feeding data generated by the two different cooperative localization approaches to different lateration algorithms – in short, to evaluate the different combinations of localization strategy and lateration algorithm.

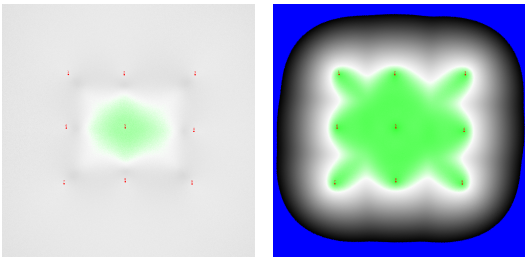
Figure 2 shows a comparison of different combinations of localization strategy and lateration algorithm. It is immediately obvious that network multilateration on the whole achieves more consistent results, i.e. results that are less dependant on choice of lateration algorithm. Classic trilateration is not at all affected by choice of localization strategy, implying that the most significant differences lie in the quality of the data used for refining the position estimate in the multilateration algorithms. NLLS is also not significantly influenced by the localization strategy, achieving consistently good results in either scenario. LLS, however, benefits greatly from network multilateration as opposed to successive refinement. AML is also improved by network multilateration, but not as drastically as LLS.

D. Interpretation

In order to analyze the difference in position estimation quality in the two different approaches, we must analyze the error sources inherent in these approaches.

The most significant error in network multilateration is the ranging error. This is due to the fact that this approach uses not only the range to its neighbors, but whole multihop paths to the anchors, each consisting of several range measurements, each in turn error-prone. Simply speaking, the expected ranging error increases with the distance in hops from a given anchor. Successive refinement, on the other hand, has its main error source in the fact that it bases its estimates on positions that are themselves estimates. This means that the further away a node is from an anchor, the more fuzzy the estimate becomes. The basic question is thus this: Which error source — ranging or position – has a stronger influence on the result of a given lateration algorithm?

Looking at our results, the answer is obvious: The ranging error has significantly lower influence on the overall error. The explanation for this observation can be found in the phenomenon of clustering that arises in scenarios with mobility: When using recursively estimated positions instead of paths to anchors, we effectively laterate using positions that are themselves estimates, instead of using positions we know to be accurate, only with estimated ranges. Considering a group of nodes with no nearby anchors, it is easy to construct a scenario where these nodes continuously estimate their own locations using the estimates of their neighbors — none of which can reach an anchor. It is obvious that in such a scenario, the position error is unbounded and will quickly render all estimation useless. Using paths to anchors with known, fixed positions will place an upper bound on the error



(a) NLLS (b) Min-Max

Fig. 3. Spatial distribution of the average position error with a dense anchor distribution of nine anchors.

propagated through the recursion steps, effectively avoiding cluster formation.

Another angle that needs to be evaluated is that of spatial distribution. Will et al. have presented a lateration algorithm simulation and visualization engine called LS^2 [11] which can visually evaluate and compare distance based lateration algorithms by displaying the spatial position error distribution. Using the images generated by LS^2, we can not only analyze the overall error of the two approaches, but also their spatial distribution.

In Fig. 3, we simulated the spatial distribution of the position error for two different algorithms. The images show the simulation error and the color of each position represents the average position error for 1000 simulation runs. The green area indicates a position error that is lower than the expected distance measurement error; the darker the color, the lower the error. The gray area indicates a position error that is higher than the expected distance measurement error; the darker the color, the higher the error. In the blue area, the error is very high and is cut to keep a good image contrast. For this example we simulated two well known algorithms: multilateration with a nonlinear least squares (NLLS) solver [12] and Min-Max [13], [14]. The average position error for NLLS is ways lower than the position error for Min-Max, but inside the convex hull of the anchors Min-Max has a much better performance than NLLS. If we were to use Min-Max for cooperative localization with successive refinement the chance is very high for each hop that we hit a non-green area, so the resulting position error would be much higher than the accumulated distance measurement error. Network multilateration faces this problem only once, because the lateration algorithm is calculated only once. NLLS shows a very homogeneous spatial distribution of the position error: nearly the whole simulation area performs around the expected value of the distance error. So NLLS should be usable for both cooperative strategies. It can be seen in Fig. 2 that this assumed behavior could be reproduced in our simulation results.

IV. Conclusion

We presented an evaluation of the two main cooperative localization approaches and analyzed their sources of position error. We showed in our evaluation that network multilateration has some advantages over successive refinement that have not been researched very well. Especially the impact of the spatial distribution on the position error needs further research. We showed that formation of clusters, which can be a major source of error in the successive refinement approach, can be effectively minimized when using network multilateration. Future work should also address the possibility to switch the lateration algorithm regarding the spatial error distribution. It is also conceivable to develop a metric based on the spatial error distribution for each hop in the network, which gives a clue as to whether to use network multilateration or successive refinement.

References

[1] N. Patwari, J. Ash, S. Kyperountas, A. Hero III, R. Moses, and N. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *Signal Processing Magazine, IEEE*, vol. 22, no. 4, pp. 54–69, 2005.

[2] D. Niculescu and B. Nath, "Ad hoc positioning system (aps)," in *Global Telecommunications Conference, 2001. GLOBECOM'01. IEEE*, vol. 5. Ieee, 2001, pp. 2926–2931.

[3] A. Savvides, H. Park, and M. Srivastava, "The bits and flops of the n-hop multilateration primitive for node localization problems," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. ACM, 2002, pp. 112–121.

[4] J. Albowicz, A. Chen, and L. Zhang, "Recursive position estimation in sensor networks," in *Network Protocols Ninth International Conference on ICNP 2001*. IEEE, 2001, pp. 35–41.

[5] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proceedings of the 7th annual international conference on Mobile computing and networking*, ser. MobiCom '01. New York, NY, USA: ACM, 2001, pp. 166–179. [Online]. Available: http://doi.acm.org/10.1145/381677.381693

[6] C. Savarese, J. Rabaey, and J. Beutel, "Location in distributed ad-hoc wireless sensor networks," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, vol. 4. IEEE, 2001, pp. 2037–2040.

[7] A. Savvides, W. Garber, R. Moses, and M. Srivastava, "An analysis of error inducing parameters in multihop sensor node localization," *Mobile Computing, IEEE Transactions on*, vol. 4, no. 6, pp. 567–577, 2005.

[8] K. Whitehouse and D. Culler, "A robustness analysis of multi-hop ranging-based localization approximations," in *Proc. 5th international conference on Information processing in sensor networks*. ACM, 2006, pp. 317–325.

[9] R. Barr, "Jist-java in simulation time users guide," 2004.

[10] ——, "Swans scalable wireless ad hoc network simulator," *User Guide (March 19, 2004)*, 2004.

[11] H. Will, T. Hillebrandt, and M. Kyas, "The fu berlin parallel lateration-algorithm simulation and visualization engine," in *Proc. 9 Workshop on Positioning, Navigation and Communication 2012 (WPNC12)*, 2012.

[12] J. E. Dennis, Jr. and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations (Classics in Applied Mathematics, 16)*. Soc for Industrial & Applied Math, 1996.

[13] A. Savvides, H. Park, and M. B. Srivastava, "The bits and flops of the n-hop multilateration primitive for node localization problems," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, ser. WSNA '02. New York, NY, USA: ACM, 2002, pp. 112–121. [Online]. Available: http://doi.acm.org/10.1145/570738.570755

[14] K. Langendoen and N. Reijers, "Distributed localization in wireless sensor networks: a quantitative comparison," *Comput. Netw.*, vol. 43, no. 4, pp. 499–518, Nov. 2003. [Online]. Available: http://dx.doi.org/10.1016/S1389-1286(03)00356-6