Range-Based Localization in Mobile Sensor Networks

Bram Dil[1], Stefan Dulman[2], and Paul Havinga[1,2]

[1] Embedded Systems, University of Twente, The Netherlands
[2] Ambient Systems, The Netherlands
b.j.dil@student.utwente.nl, havinga@cs.utwente.nl,
dulman@ambient-systems.net

Abstract. Localization schemes for wireless sensor networks can be classified as range-based or range-free. They differ in the information used for localization. Range-based methods use range measurements, while range-free techniques only use the content of the messages. None of the existing algorithms evaluate both types of information. Most of the localization schemes do not consider mobility. In this paper, a Sequential Monte Carlo Localization Method is introduced that uses both types of information as well as mobility to obtain accurate position estimations, even when high range measurement errors are present in the network and unpredictable movements of the nodes occur. We test our algorithm in various environmental settings and compare it to other known localization algorithms. The simulations show that our algorithm outperforms these known range-oriented and range-free algorithms for both static and dynamic networks. Localization improvements range from 12% to 49% in a wide range of conditions.

1 Introduction

A wireless sensor network is a network where small sensors with limited hardware capabilities communicate wirelessly with each other. First all nodes are placed in a random matter (like dropping them from an airplane). "When the nodes are dropped", they are capable of communicating with each other within a certain communication radius. The network can be considered as an undirected graph using its connectivity and range measurement information. When certain information is propagated through the network, nodes can be located by using that information.

Wireless sensor networks hold the promise of many new applications in the area of monitoring and control. Examples include target tracking, intrusion detection, wildlife habitat monitoring, climate control, and disaster management ([3]). Localization of the nodes is one of the main issues in a wireless sensor network. While many algorithms have been proposed to estimate the position of the nodes, there is still no algorithm that performs best in all networks.

As range measurements between nodes contain some error, the nodes' locations can only be estimated. This is called "the range error" problem ([2]). Node

K. Römer, H. Karl, and F. Mattern (Eds.): EWSN 2006, LNCS 3868, pp. 164–179, 2006.

localization algorithms dependent on range measurements are sensitive to range errors. While range-free algorithms overcome this problem, they perform badly in irregular networks ([3], [4]). Several studies have been performed to minimize the impact of these range measurement errors. These studies estimate a node's position by giving a certain weight to each measurement or estimated node's location. These weights are then used to compute a least square solution by using an Iterative Weighted Least Square Method (IWLSM) ([2], [7], [19]).

In general, localization algorithms follow the following scheme ([3]): anchor-unknown distance determination, deriving a node's position given the anchor distances, and then refinement of the position estimates. Because mobile sensor networks are changing fast as time progresses, not much effort has been invested in researching the refinement phase. However, this phase can be successfully applied to static networks ([1], [2]).

Most of the current proposed localization algorithms apply an Iterative Weighted Least Square Method ([1], [2], [6], [7], [18], [19]). They differ in determination of the anchor-unknown distances and in weights used in the IWLSM.

Improved MDS-MAP ([5]) uses a different technique: Multi-Dimensional Scaling (MDS). This method uses all available local information around a node and computes a local map for each node. By merging these local maps and known anchor positions, a global map can be computed. With this global map available, the nodes can estimate their position. This centralized localization technique uses a lot of communication and is therefore not applicable in mobile sensor networks. In this study, we compare our algorithm with an IWLSM, using the same weights as our localization scheme. These weights are based on standard and available knowledge of the accuracy of the range measurement hardware.

In addition, we compare our algorithm with the following range-free algorithms: DV-hop ([18]) and an SMCL method ([13]). With increasing range measurement error, the positioning error increases of the range-based algorithms. We made a comparison with range-free algorithms when high range measurement errors are present in the network.

Our algorithm adapts a Monte Carlo Localization (MCL) method, which has been successfully implemented in robotics localization ([11], [12]) and range-free localization in a mobile wireless sensor networks ([13]). Our Monte Carlo Localization algorithm combines range-free and range-based information to improve its performance. It uses the range-free information to increase its robustness even when high range measurement errors are present in the network. In addition, it improves the localization accuracy and lowers the computational costs by using the range measurements. In addition, we use the mobility in the network to increase accuracy.

This paper is organized as follows: After the problem formulation in Section 2, Section 3 describes a known Sequential Monte Carlo Localization solution in a range-free mobile wireless sensor network. In section 4, we introduce our new algorithm that uses all of the range-free information described in Section 3. Section 5 presents the simulation reports and comparisons with other localization

algorithms. In Section 6, we analyze the results of our algorithm and compare them with the results of other localization algorithms. Section 7 summarizes the conclusions.

2 Problem Formulation

In a mobile sensor network, we assume that the time is divided into fixed constant time units. In a time unit, the node moves away from its previous location to its current location. When a time unit has elapsed, the localization algorithm has to locate the unknowns with the information available. Our algorithm is interested in estimating the filtering distribution of a node when range measurements are available in the network. The Sequential Monte Carlo approach provides simulation based solutions to estimate the posterior distribution of nonlinear discrete time dynamic models ([9]).

We formulate the mobile localization problem in a state space form as follows: Let t be the discrete time given in time units; l_t is the position distribution of a node given at time t; o_t represents the observations of a node received from the anchors between time $t-1$ and t. We are interested in estimating recursively in time the so-called filtering distribution: $P(l_t|o_{0:t})$. The filtering distribution is represented by N weighted samples which are updated every time unit, using an importance function. The performance of the Sequential Monte Carlo Localization algorithm is highly dependent on this latter distribution function. In the ideal case samples are drawn from the posterior distribution: $P(l_{0:t}|o_{0:t})$, but most of the time it is impossible to sample directly from the posterior distribution. The general Sequential Monte Carlo method looks like ([15]):

$$P(l_{0:t}|o_{0:t}) = P(l_0|o_0) \prod_{k=1}^{t} P(l_k|l_{0:(k-1)}, o_{0:k}) \tag{1}$$

Different importance functions have been proposed through the years, which are most of the time special cases of this general algorithm ([15]). An overview of Sequential Monte Carlo methods can be found in ([15]).

3 Known SMCL Solution

A Sequential Monte Carlo Localization algorithm for mobile wireless networks is described in [13]. In this article no range measurements are present so the observations of the node (o_t) only consists of anchor positions. Connectivity constraints are constructed from these anchor positions. They use a prior importance function ([17]), which implies that the importance function draws samples without any knowledge of the observations. They use the following recursive function:

$$P(l_t|o_{0:t}) = P(l_0) \prod_{k=1}^{t} P(l_k|l_{k-1}) \tag{2}$$

$$\widetilde{w}_t^{(i)} = \widetilde{w}_{t-1}^{(i)} P(o_t|l_t^{(i)}) \tag{3}$$

Their algorithm is divided into three phases which are described in the next three subparagraphs. In the last subparagraph we discuss the observations and extensions.

3.1 Prediction Phase

In the prediction phase the samples are drawn from the previous predictions: $P(l_t|l_{t-1})$. The algorithm assumes that nodes know their maximum speed V_{max}. Given a previous position l_{t-1} and speed constraint V_{max}, possible predicted positions by l_{t-1} are within a circular region with origin l_{t-1} and radius V_{max}. This gives the following constraint:

$$P(l_t|l_{t-1}) = \begin{cases} \frac{1}{\pi V_{max}^2} & \text{if } d(l_t, l_{t-1}) \leq V_{max}, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Here $d(l_t, l_{t-1})$ is the distance between the current prediction l_t and the previous prediction l_{t-1}. Our algorithm also uses this speed constraint (Section 4.2).

3.2 Filtering Phase

In the filtering phase the predictions that do not lie within the connectivity constraints are filtered (Equation 3: $P(o_t|l_i)$). Because the transmission range is modelled as a perfect circle and only two-hop away information is available, the following condition holds:

$$\text{filter}(p) = \forall a \in S, d(p,a) \leq tr \cap \forall b \in T, tr \leq d(p,b) \leq 2tr \tag{5}$$

Here is p the prediction; S is the set of one-hop away anchors, T is the set of two-hop away anchors; tr is the transmission range and $d(p,a)$ is the distance between prediction p and anchor a. Because $P(o_t|l_i)$ (Equation 3) can only be 1 or 0, the weights associated with the predictions are also 1 (valid) or 0 (invalid). Our algorithm uses an extended version of this filtering condition (Section 4.2).

3.3 Re-sampling

After one prediction and filtering step, the number of valid predictions is of variable size. To keep the number of valid predictions of constant size, the process of predicting and filtering is repeated until N valid samples are drawn. The simulations proved that $N = 50$ was sufficient ([13]). The final position estimate is the mean of the predictions.

3.4 Observations and Extensions

In the first time unit, no previous predictions are available ($P(l_0)$). In [13], these previous predictions are placed randomly in the possible area. Placing the previous predictions randomly gives poor results in the first few time units.

We propose that if no previous predictions are available, the algorithm makes predictions based upon the first connectivity constraint received: $P(l_1|o_1^{(1)})$. This proposal, shortly mentioned in [13], not only improves the results in the first few time units but also decreases the "initialization phase" time. We then use the following recursive function:

$$P(l_t|o_{0:t}) = P(l_1|o_1^{(1)}) \prod_{k=2}^{t} P(l_k|l_{k-1}) \tag{6}$$

The importance function draws predictions based only on previous predictions. This means that if the constraints based upon connectivity and previous predictions are tight, many predictions have to be made to come to N valid predictions. It is even possible that this algorithm cannot make any valid predictions. That is why we use a looping limit that limits the number of times the process of predicting and filtering is done. When the looping limit is reached and no valid predictions are made, the algorithm makes predictions as if it had no previous predictions.

4 SMCL and Range Measurements

In this section, we discuss the case when range measurements are present in the network. When we include the range measurements into the recursive Sequential Monte Carlo computation, we obtain the following filtering distribution:

$$P(l_t|o_{cc,0:t}, o_{rm,0:t}) \tag{7}$$

Here $o_{cc,0:t}$ are the connectivity constraints and $o_{rm,0:t}$ are the range measurements.

We made an approximation of the optimal solution ([10]) by dividing the optimal solution into several suboptimal solutions as the optimal solution cannot be evaluated directly. We propose the following new recursive computation:

$$P(l_t|o_{cc,0:t}, o_{rm,0:t}) \approx \sum_{o_{rm,1}^{(i)} \in o_{rm,1}} P(l_1|o_{cc,1}, o_{rm,1}^{(i)}) \prod_{k=2}^{t} \sum_{o_{rm,k}^{(i)} \in o_{rm,k}} P(l_k|l_{k-1}, o_{cc,k}, o_{rm,k}^{(i)}) \tag{8}$$

Figure 1 shows an overview in pseudocode of the algorithm. The nodes locally use this algorithm to estimate their positions with the received information of the anchors. The different phases of the algorithm are discussed in the following subsections.

```
FOR EVERY "time unit" DO
   saved predictions become saved previous predictions      (section 4.2)
   receive, save and forward anchor-unknown distances       (section 4.1)
   FOR ALL "received range measurements" DO
      Compute Nlocal                                        (section 4.3)
      Predict and save Nlocal predictions                   (section 4.2)
   END FOR;
   Compute weight for each saved prediction                 (section 4.4)
   Compute final position                                   (section 4.4)
END FOR;
```

Fig. 1. Pseudocode of the range-based SMCL algorithm

4.1 Anchor-Unknown Distances

In this phase, the nodes determine their distance to one or multiple anchors by using different range measurements. This information is needed by our algorithm to localize the nodes. Our algorithm uses sum-dist to determine this distance, nameless in [1] and later named in [3]:

First, anchors start flooding the network with their position, a hop distance and path length set to zero. Each receiving node adds the measured range to the path length and increases the hop distance by one. Only the shortest hop distance and path length is forwarded for each anchor. At the end of this phase, every node has stored the anchor positions, minimum hop distances, and minimum path lengths to several anchors.

4.2 Prediction Phase

In the prediction phase, samples are drawn from the previous predictions, connectivity constraints and one range measurement:

$$P(l_k|l_{k-1}, o_{cc,k}, o^{(i)}_{rm,k}) \tag{9}$$

Our algorithm assumes that the nodes know their maximum speed V_max (Section 3.1, Equation 4), and the filtering condition that represent the connectivity constraints (Section 3.2, Equation 5) is updated to support n-hop away anchors:

$$\text{filter}(p) = \forall a \in S, d(p,a) \leq tr \cap \forall b \in T, tr \leq d(p,b) \leq n \cdot tr \tag{10}$$

Here is T the set of n-hop away anchors, where $n \geq 2$. We also assume that the transmission range is a perfect circle.

The algorithm needs to evaluate one extra observation compared to the other SMCL scheme: the range measurement. By using standard geometry, we can easily evaluate Equation 9. We must do this for all range measurements. We use this construction because the optimal solution cannot be evaluated directly. An approximation is made by dividing the optimal solution into several suboptimal solutions:

$$P(l_t|l_{t-1}, o_{cc,t}, o_{rm,t}) \approx \sum_{o^{(i)}_{rm,t} \in o_{rm,t}} P(l_t|l_{t-1}, o_{cc,t}, o^{(i)}_{rm,t}) \tag{11}$$

In this case, every range measurement can be seen as a sampling function, not considering the other range measurements. Given range measurement rm to anchor position a, the predictions according to the range measurement are somewhere located at the edge of the circle with origin a and radius rm. This gives the following constraint:

$$P(l_t|o_{rm}) = \begin{cases} 1 & \text{if } d(l_t, a) = rm, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

Note that after the prediction phase we only have valid predictions, so we do not need a filtering phase.

4.3 Weights and Sample Size

Our algorithm uses a constant number of predictions: N. This is done to keep the computational costs at a low and constant level. In the prediction phase, the sampling of the predictions is divided into several sampling functions by the range measurements. So every sampling function samples a portion of N predictions: N_{local}. The size of N_{local} depends on the precision of the range measurement, formulated as: $\frac{1}{\sigma^2_{rm}}$. σ^2_{rm} stands for the variance of the range measurement. This variance is based on the hop distance associated with the range measurement. Every range measurement consists of "hop count" independent range measurements. We use an approximation made in [19]:

$$\sigma^2_{rm,i} \approx i \cdot \sigma^2_{rm,1} \tag{13}$$

Here i stands for the number of hops. Our algorithm uses this approximation to compute the ratio between the precisions of the various range measurements. This ratio is used to compute the size of N_{local}, and is later used to compute the final position estimation. Note that this ratio can be calculated only using the "hop count" because we assume that the nodes have the same distance measurement hardware ($\sigma^2_{rm,1}$ is a constant).

4.4 Computing the Final Position Estimation

In this phase, the algorithm uses the predictions, made in the prediction phase, to compute its position estimation. With the available range measurements and associated weights (Equation 13), a weight is estimated for each prediction. Let $p_1 \ldots p_i$ be all predictions with locations $x_1, y_1 \ldots x_i, y_i$. Let $A_1 \ldots A_j$ be all known anchor positions with associated range measurements $R_1 \ldots R_j$ to the specific node. The weights of the predictions are computed by the summed squared error multiplied by the appropriate range measurement weights:

$$\sigma^2_{p_i} = \sum_{k=1}^{j} \frac{1}{\sigma^2_{rm,k}} \Big(d(p_i, A_k) - R_k \Big)^2 \tag{14}$$

$\frac{1}{\sigma^2_{rm,k}}$ stands for the precision estimate of range measurement R_k. We take the summed square error as an estimate of the variance of prediction p_i: $\sigma^2_{p_i}$. Using

this estimate, the precision of prediction p_i is $\frac{1}{\sigma^2_{p_i}}$. If we have N predictions then the optimal position (x, y) is where: $\sum_{i=1}^{N} \frac{1}{\sigma^2_{p_i}} \Big((x_i - x)^2 + (y_i - y)^2 \Big)$ is minimized. This weighted least square solution can be computed with an iterative weighted least square method. It can also be computed with a weighted mean method. The weighted mean method uses less computation power and is therefore a good replacement:

$$\begin{aligned} x &= \frac{1}{\sum_{i=1}^{N} \sigma_{p_i}^{-2}} \cdot \sum_{i=1}^{N} \sigma_{p_i}^{-2} \cdot x_i \\ y &= \frac{1}{\sum_{i=1}^{N} \sigma_{p_i}^{-2}} \cdot \sum_{i=1}^{N} \sigma_{p_i}^{-2} \cdot y_i \end{aligned} \tag{15}$$

This algorithm uses the ratio between the weights to compute the weighted least square solution.

4.5 Observations

It is possible that no valid predictions can be made. In this case, the algorithm makes predictions as if it had no previous predictions. When the range measurements are really bad, it is even possible that our algorithm cannot make any valid predictions with no speed constraints. In that case, the recursive function proposed in Section 3.4 is used (Equation 6).

A range measurement that does not satisfy its own connectivity constraint changes its value to the nearest number that satisfies this constraint. This increases the performance of the algorithm in several ways:

- More predictions can be made, giving a better representation of the position distribution.
- Peaks in the range measurements have less influence on the final position estimation.

5 Simulations

In this section, we analyze our algorithm by running several simulations using MatLab. In these simulations, we test our algorithm with different values for algorithm-specific parameters and under various environmental settings. In addition, we compare our results to other localization techniques: these are the IWLSM, the range-free MCL scheme ([13]) and DV-HOP ([18]). We analyze the results of the localization schemes by looking at the mean error versus the communication costs.

5.1 General Simulation Set-Up

Except when stated otherwise, we use the same general set-up for all simulations. The sensor nodes are uniformly placed within a $1x1$ units^2 area and a transmission range tr of 0.125 units is used. For simplicity, the transmission range is simulated as a perfect circle and messages are always received correctly.

The parameters we vary are:

- The number of predictions drawn by the sampling function. In general, we use a number of 50 samples.
- The number of nodes placed within the area. In general, we use a number of 180 nodes. The node density (average number of 1-hop away nodes) is determined by simulation. The general set-up has a node density of: 13.9.
- The number of anchors placed within the area. In general, we use a number of 20 anchors. The anchor density (average number of 1-hop away anchors) is determined by simulation. The general set-up has an anchor density of: 1.3.
- The speed of the nodes, which we choose randomly from $[0, Vmax]$. The nodes' speed is given as a ratio of the transmission range. In general, we use a speed of $[0, 1]$.
- The Time-To-Live (TTL) of the messages. This value indicates the number of times a message is forwarded. We keep the communication costs per algorithm the same with this parameter, so the performance of the different algorithms are determined by the localization error. DV-Hop isn't affected by the TTL and has different communication costs than the other algorithms. In general, we use a TTL of 4 for every algorithm.
- The range measurement errors, which we simulate by a Gaussian distribution with the real distance as the mean. The standard deviation of the error is represented as a ratio of the real range. In general, we use a standard deviation of 0.2 ([7]). This value is based on the picoradio ([8]) that uses Received Signal Strength Indication (RSSI) for range measurements.
- We tested each simulation setup for 50 runs, each consisting of 50 time units.

We adopt a modified ([13]) random waypoint mobility model ([16]) for the nodes. With this model, a node randomly chooses its destination. After arriving at its destination, the node chooses a new destination. Furthermore, the speed of the nodes are changed and randomly chosen from $[0, Vmax]$ after each time unit and when the nodes arrive at their destinations. We use this model to maintain an average speed. Before localization, we run the modified random waypoint mobility model for several time units to maintain the distribution created by this mobility model.

We use the following settings for the other localization schemes:

- The extensions proposed in this article (Section 3.4 and Equation 6) are used for the range-free SMCL scheme ([13]). We also need another update to support different values of TTL (Section 4.2, Equation 10). We use a looping limit of 10 and a sample size of 50, which should be enough according to [13].
- The IWLSM uses the same weights and range measurement values as our algorithm.
- We use the DV-hop localization scheme as proposed in ([18]).

5.2 Accuracy

In this section, we analyze the performance of the various algorithms in case of the general settings, described in section 5.1. Figure 2 shows the mean error as a

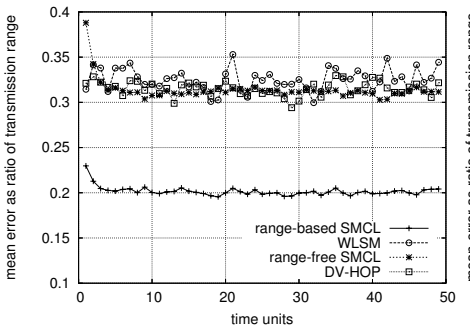

Fig. 2. Mean error per time unit

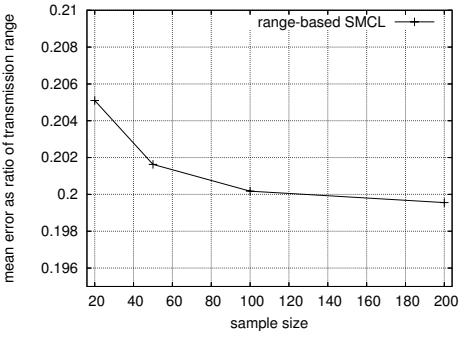

Fig. 3. Mean error per sample size

ratio of the transmission range for all the nodes that received information from three or more anchors. For both of the SMCL schemes, the localization process can be divided into the initialization phase and the stable phase ([13]). In the initialization phase, the positioning error decreases rapidly as new observations are evaluated. The region of the position distribution becomes smaller until a stable phase is reached. In the stable phase, the impact of the mobility and connectivity constraints on the position distribution remain in equilibrium, and the mean error fluctuates around some mean value. From the first to the last time unit our SMCL scheme outperforms the other algorithms in terms of the mean error. All the algorithms have the same communication costs (TTL), except for DV-hop that has a higher communication cost.

The localization error in the first time unit gives an indication of the mean error in a static network. The simulations show an improvement of the mean error in the dynamic case of 36% and in the static case of 27%.

5.3 Sample Size

The predictions and associated weights are a representation of the probability distribution. When more predictions are made, a better approximation of the probability distribution is made. While maintaining more samples improves accuracy, it also increases the computational and memory costs for the node. In this section we try to find a balance between the benefits and losses.

Figure 3 shows that increasing the sample size beyond 20 has a minimal effect on the positioning error with these specific simulation settings. Choosing the right sample size mainly depends on the quality and quantity of the received observations per node. This quality and/or quantity is directly influenced by the TTL of a message, seed density and precision of the range measurements. So more samples are needed when the accuracy of the received observations increases.

5.4 Message TTL

The increase of the number of times a message is forwarded is equivalent to an increase of the average amount of information received by a node. Using this amount, we distinguished two types of nodes:

- Good connected nodes, that receive information from three or more anchors.
- Bad connected nodes, that receive information from two or less anchors.

These two types of nodes are not only divided by connectivity to anchors, but also by localization error. Therefore, the ratio between the number of good and bad connected nodes mainly determines the overall mean error. This ratio is affected by the TTL. Figure 5 shows that increasing the TTL, decreases the overall mean error rapidly. The drawback is that increasing the TTL, increases the communication costs. Figure 6 shows what affect the TTL has on the average number of messages sent by a node. After each TTL wave the received information is combined and forwarded in one message, so this average number of messages represents the minimum. "Other algorithms" in Figure 6 and 7 represent the range-based, range-free and IWLSM localization algorithms, because the communication costs for these algorithms are equal. The communication costs of the DV-hop algorithm cannot be changed by the TTL, because it consists of two predefined phases. Figure 6 shows that the communication costs for the DV-Hop algorithm are more than twice as big as for the other algorithms.

Hence, every time a message is forwarded, the range measurements become less precise and the connectivity constraints become less tight. This is illustrated

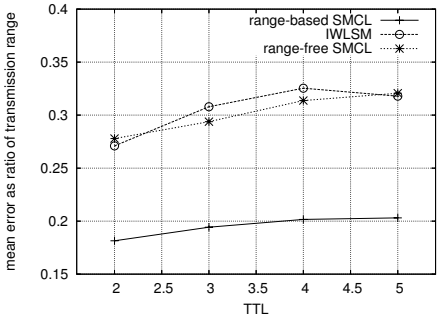

Fig. 4. Mean error per TTL, good connected nodes

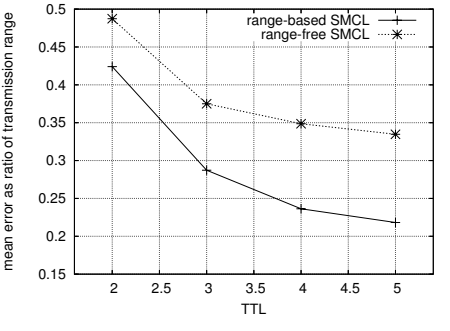

Fig. 5. Mean error per TTL, good and bad connected nodes

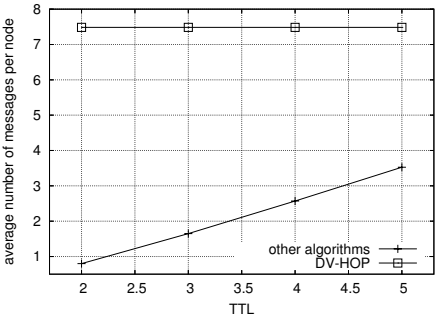

Fig. 6. Number of messages per TTL

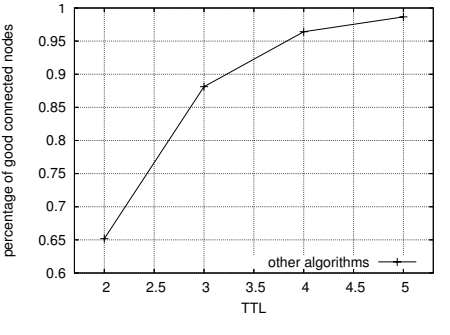

Fig. 7. Good connected nodes per TTL

by Figure 4, which shows a decrease of accuracy of good connected nodes with increasing TTL. Figure 4 also shows that this decrease of accuracy with increasing TTL is minimal.

The bad connected nodes have a dramatically high mean error compared to the good connected nodes. So including the position estimates of the bad connected nodes is questionable. When these position estimates are not included, the chosen TTL for a mobile wireless sensor network mainly depends on the desired localization coverage. Figure 7 shows the increase of good connected nodes when the TTL increases.

Therefor, in this paper we use the good connected nodes for the determination of the mean error.

5.5 Anchor Density

Increasing the number of anchors in the network increases the average number of observations per node. The localization accuracy depends on this number. The drawback is that deploying more anchors increases the network and deployment costs. Figure 8 shows the effect of anchor density on the mean error. The number of 10, 20 and 30 anchors represent the following anchor densities: 0.65, 1.3 and 1.9. Our algorithm performs in terms of mean error 31% better with low anchor density and 33% with high anchor density than the best other algorithm.

5.6 Node Density

Figure 9 shows the effect of the node density on the localization accuracy. The number of 100, 150, 200, 250 and 300 nodes represent the following node densities: 7.4, 10.6, 13.9, 17.0 and 20.1.

We only need the hop-distance to several anchors for the range-free SMCL scheme to work. That is why this algorithm only requires a low node density to run properly ([13]). Increasing the node density slowly increases the number of observations per node, so the mean error of the range-free SMCL scheme remains practically stable while the node density is changed. We know that range-based algorithms, especially the performance of sum-dist, are affected by the node

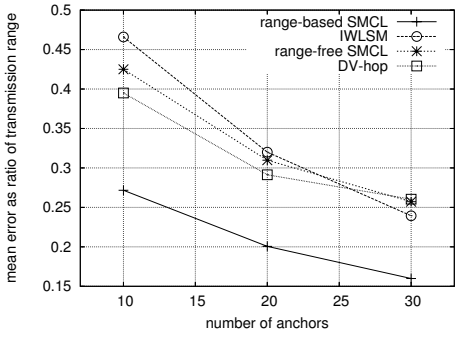

Fig. 8. Mean error per anchor density

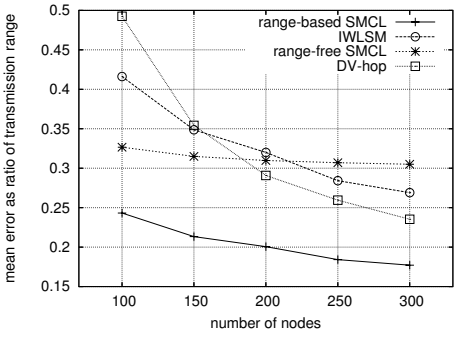

Fig. 9. Mean error per node density

density ([3]). Increasing the node density leads to straighter paths between the nodes and anchors, so that the shortest path becomes a better approximation of the real distance. When the real distances are better approximated, the range-based algorithms perform better.

Figure 9 shows an improvement of the mean error of 25% with low node density and 25% with high node density than the best other algorithm compared with our algorithm.

5.7 Node Speed

The frequency of the localization announcements influences the traveled distances of the nodes per time unit. Therefore, if localization announcements are more frequent, the speed of the node per time unit decreases. Updating the position estimate more frequently improves localization accuracy at the cost of communication.

Increasing the speed, increases the size of the prediction surface, which is constructed by the speed and connectivity constraints. This size affects the localization error, because it limits the position distribution.

In this simulation set-up, we only compare our algorithm with its range-free counterpart because the node speed is only used by the SMCL schemes. Figure 10 shows that our algorithm performs in terms of mean error 36% better with a low speed and 49% with a high speed than the range-free MCL scheme.

5.8 Range Measurement Error

The precision of the range measurements has a significant influence on the accuracy of the range-based localization schemes.

Figure 11 shows that our algorithm performs 40% better with a low range measurement error and 41% with a high range measurement error than the IWLSM. Even with a range measurement error of 0.4 our algorithm has a 12% lower mean error than the best range-free algorithm.

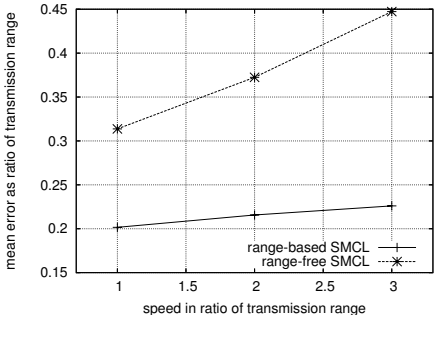

Fig. 10. Influence of speed on mean error

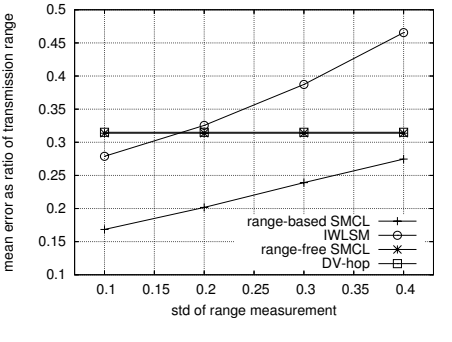

Fig. 11. Influence of range measurement error on mean error

6 Analysis

In this section, we analyze our algorithm and compare it with other known algorithms. Our algorithm is interested in the filtering condition (Equation 7). We made an approximation of the optimal solution by using several suboptimal solutions (Equation 11) as the optimal solution cannot be evaluated directly. In our case, we use the range measurements to characterize these suboptimal solutions. We use the same range measurements in the final position estimation. This construction has several benefits:

- Bad range measurements filtered by the connectivity and/or speed constraints do not have any influence on the position distribution.
- Every range measurement gives an indication of the real position of the node, so that all measurements are evaluated in the final position estimation.
- The performance of our algorithm is less dependent on the range measurement errors (Figure 11).

The range-free MCL scheme([13]) uses a prior sampling function. This means that this MCL scheme makes predictions based upon its previous predictions and filters bad predictions with the connectivity constraints. This iterative two-step construction is needed because the posterior distribution cannot be evaluated directly. Our range-based algorithm uses a non-iterative MCL scheme and evaluates as much information as possible in the sampling function. This has several benefits over its range-free counterpart:

- Every previous prediction has a more equal chance to make a prediction.
- The computational costs are more constant and less situation dependent.
- Our algorithm is less dependent on the speed of the node (Figure 10).
- Using the range measurements, our algorithm decreases localization error by 12% to 49% (Figure 11).

Many of the known range-based and range-free localization algorithms use an Iterative Weighted Least Square Method to estimate positions. The IWLSM starts from an initial estimation and improves the position until the improvement is smaller than a certain value. The position estimate is the global or local minimum of the summed weighted squared error, using the range measurements and associated weights. In most situations, especially when the range error increases, there are more local minima. Dependent on the starting position the IWLSM chooses one of these local minima. Our approach uses the fact that these local minima are located near the range measurements. Our algorithm tries to make predictions near these local minima to evaluate the entire surface, while not picking one of these local minima.

As an example, consider the case when the positions of the anchors are collinear and the range measurements contain no error. In the perfect case, the IWLSM chooses one of the local minima, while our algorithm chooses the mean of the two local minima. If we also evaluate the previous position distribution, the mobility and connectivity constraints the collinearity problem [1] is often solved.

In this paper, the communication costs for the different algorithms are equal, except for the DV-hop algorithm that uses much more communication (Figure 6).

7 Conclusions

In this paper, we proposed a non-iterative MCL scheme that uses all information to improve localization accuracy and robustness. This information consists of range measurements, connectivity constraints, and mobility information. Simulations show that our algorithm decreases the localization error by 12% to 49% in static and dynamic networks under a wide range of conditions, even when the node and network resources are limited. Future work aims at testing our algorithm in other mobility models and real life settings.

References

1. A. Savvides, H. Park and M. Srivastava: The Bits and Flops of the N-Hop Multilateration Primitive for Node Localization Problems. In First ACM International Workshop on Wireless Sensor Networks and Application, Atlanta, GA, September 2002.
2. C. Savarese, J. Rabay and K. Langendoen: Robust Positioning Algorithms for Distributed Ad-Hoc Wireless Sensor Networks. USENIX Technical Annual Conference, Monterey, CA, June 2002.
3. Koen Langendoen and Niels Reijers: Distributed localization in wireless sensor networks: A quantitative comparison. In Computer Networks (Elsevier), special issue on Wireless Sensor Networks, 2003.
4. Y. Shang, W. Ruml, Y. Zhang and M. Fromherz: Localization From Mere Connectivity. MobiHoc'03, Annapolis, Maryland, June 2003.
5. Yi Shang and Wheeler Ruml: Improved MDS-based localization. In Infocom 2004
6. L.Evers,W.Bach, D.Dam,M.Jonker, H.Scholten, and P.Havinga: An iterative quality based localization algorithm for adhoc networks. In Department of Computer Science, University of Twente, 2002.
7. L.Evers, S.Dulman, P.Havinga: A distributed precision based localization algorithm for ad-hoc networks. Proceedings of Pervasive Computing (PERVASIVE 2004).
8. Jan Beutel: Geolocation in a picoradio environment. In MS Thesis, ETH Zurich, Electronics Lab, 1999.
9. J.E.Handschin: Monte Carlo Techniques for Prediction and Filtering of Non-Linear Stochastic Processes. Automatica 6. pp. 555-563. 1970.
10. V.S.Zaritskii, V.S.Svetnik, L.I.Shimelevich: Monte Carlo technique in problems of optimal data processing. Automation and Remote Control 12: 95-103. 1974.
11. F.Dellaert, D.Fox, W.Burgard, S.Thrun: Monte Carlo Localization for Mobile Robots. IEEE International Conference on Robotics and Automation (ICRA). May 1999.
12. S.Thrun, D.Fox, W.Burgard, F.Dellaert: Robust Monte Carlo Localization for Mobile Robots. Artificial Intelligence Journal. 2001.
13. L.Hu, D.Evans: Localization for Mobile Sensor Networks. Tenth Annual International Conference on Mobile Computing and Networking (MobiCom 2004), USA. 2004.

14. A.Kong, J.S.Liu, W.H.Wong: Sequential Imputations and Bayesian Missing Data Problems. Journal of the American Statistical Association. Volume 89, pp. 278-288. 1994.
15. A.Doucet, S.Godsill, C.Andrieu: On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. Statistics and Computing. Volum 10, pp. 197-208. 2000.
16. T.Camp, J.Boleng, V.Davies: A survey of Mobility Models for Ad Hoc Networks Research. Wireless Communications and Mobile Computing. Volume 2, Number 5. 2002.
17. H.Tanizaki, R.S.Mariano: Nonlinear and non-Gaussian statespace modeling with Monte-Carlo simulations. Journal of Econometrics 83: 263-290. 1998.
18. D.Niculescu, B.Nath: Ad hoc positioning systems. In: IEEE Globecom 2001, San Antonio. 2001.
19. S.Dulman, P.Havinga: Statistically enhanced localization schemes for randomly deployed wireless sensor networks. DEST International Workshop on Signal Processing for Sensor Networks, Australia. 2004.