Real-time Trajectory Estimation in Mobile Ad Hoc Networks

Sae Fujii Takashi Nomura Takaaki Umedu Hirozumi Yamaguchi Teruo Higashino
Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka 565-0871 Japan
Japan Science and Technology Agency, CREST
{s-fujii, t-nomura, umedu, h-yamagu, higashino} @ ist.osaka-u.ac.jp

ABSTRACT

In this paper, we propose a new trajectory estimation method named TRADE (TRAjectory estimation in DEcentralized way). TRADE is a range-free localization algorithm in fully decentralized mobile ad hoc networks. In TRADE, each mobile node periodically transmits messages containing its estimated trajectory information, and re-computes its own trajectory using those from its neighbors. This information exchange considerably contributes to improvement of the position accuracy. Furthermore, we give the optimal design of the protocol based on the analysis of the algorithm property. Through the analysis, we consider how much trajectory information should be exchanged among nodes to estimate the position within a certain error range in the protocol design. We have evaluated the position accuracy under various settings, and have shown the effectiveness of the protocol in the real world through two realistic application examples.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Distributed networks, Wireless communication*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems —*Distributed applications*

General Terms

Algorithms, Design

Keywords

trajectory estimation, mobile ad hoc network, localization, range-free, decentralized algorithm

1. INTRODUCTION

In the future society where ubiquitous devices are widely spreading, the location information of mobile devices is expected to be used for many ubiquitous services such as behavior analysis of humans, navigation, advertisement and location-aware services. Recently, many cellular phones and car navigation systems have GPS receivers, but such ubiquitous services are often provided in the situations where GPS is not available such as indoor areas, underground areas and complicated urban districts with a lot of buildings.

To provide location information to such services everywhere, location estimation using wireless communication devices of mobile nodes has been considered (see Sec. 2). In particular, "on-line" location estimation, which means real-time and distributed location estimation, is required in many situations. For example, when customers enter a huge shopping mall, they might want to know the shops they have not visited yet. Also, at stations with very complex structure, passengers may need more intuitive route navigation.

In indoor object tracking techniques, camera based methods are often used for behavior analysis of pedestrians in towns and for marketing in supermarkets (*e.g.,* [3, 10]). Since they are mainly designed not for real-time location-aware services but for trace analysis purpose, they focus on recognition and identification techniques like identifying a single object using two cameras. Also, RFID based tracking techniques like Ref. [7] have been proposed. However, the cost to deploy and manage the RFID tags is not negligible. In Ref. [4], we have proposed a method to estimate trajectories of pedestrians. It is basically a range-free "off-line" (*i.e.,* centralized) method that localizes mobile nodes using sparsely-deployed landmarks. However, such centralized techniques do not take into account computation and communication overhead in general. Also, they cannot provide real-time services such as intuitive route/shop navigation. Therefore, real-time and decentralized position estimation techniques that can work on mobile nodes would be expected.

In this paper, we propose a new real-time position estimation method named TRADE (TRAjectory estimation in DEcentralized way). Mobile nodes exchange their estimated trajectory information with each other, and use the information to update their own trajectory. This information exchange considerably contributes to the improvement of the accuracy of the trajectory estimation. TRADE is a decentralized and autonomous protocol, and is therefore robust to mobility, appearance and disappearance of nodes. The basic operation of each node in TRADE is to periodically transmit the latest trajectory information of itself (and sometimes of its neighbors) on the wireless channel. Then, based on the trajectory information received from its neighbors, each node considers the constraints on the connectivity with its neighbors, and updates its trajectory using a vector

MSWiM'09, October 26–29, 2009, Tenerife, Canary Islands, Spain.

that corrects the current position to satisfy the constraints. This update process is very lightweight because each node only computes this correction vector. By iterating this update process among neighboring nodes interactively, their estimated trajectories are gradually refined.

In addition, we give the optimal design of the protocol based on the analysis of the algorithm property. In the extreme case that each node collects complete information about all the other nodes' trajectories, the algorithm will perform as a centralized one, but need considerable amount of traffic. In order to find the truly effective information to pursue certain accuracy with reasonable amount of traffic, we give the analysis to determine how much trajectory information should be exchanged among nodes.

We have evaluated the position accuracy by simulations under various settings, and have shown the effectiveness of our protocol through two realistic application examples.

2. RELATED WORK AND CONTRIBUTION

Many localization algorithms using ad hoc communication devices have been proposed. They use different types of range measurement techniques such as RSSI and TDoA [1, 8, 12]. Meanwhile, for localization of sensor nodes, where each node has limited power supply and equipments due to hardware costs and the network has a number of such nodes, most algorithms only assume connectivity information among nodes. The same situation may happen in localizing highly dynamic mobile nodes since it is difficult to stably measure RSSI or time of signal arrival.

The localization methods are also distinguished by their form of computation, "centralized" or "decentralized". For example, MDS-MAP [13] is a centralized localization that calculates the relative positions of all the nodes based on connectivity information by Multidimensional Scaling (MDS). Similarly, DWMDS (Dynamic Weighted MDS) [2] uses movement constraints in addition to the connectivity information, and estimates the trajectories of mobile nodes. TRACKIE [4] first estimates mobile nodes that were likely to move between landmarks straightly. Based on their estimated trajectories, it estimates the trajectories of the other nodes. Since these centralized algorithms use all the information about connectivity between nodes and compute the trajectories off-line, the estimation accuracy is usually better than decentralized methods. In decentralized methods, the position of each node is computed by the node itself or cooperation with the other nodes. For example, APIT [5] assumes a set of triangles formed by landmarks, checks whether a node is located inside or outside of each triangle, and estimates its location. Amorphous [11] and REP [9] assume that location information is sent through multi-hop relay from landmarks, and each node estimates its positions based on hop counts from landmarks. In particular, REP first detects holes in an isotropic sensor network, and then estimates the distance between nodes accurately considering the holes. In MCL [6], each mobile node manages its Area of Presence (AoP) and refines its AoP whenever it encounters a landmark. In UPL[14], each mobile node estimates its AoP accurately based on AoP received from its neighboring nodes and obstacle information.

The contribution of this paper compared with the existing work is summarized as follows. (1) We provide a new decentralized method to estimate the position of each node in mobile ad hoc networks. The algorithm is fully decentralized and autonomous, and is therefore robust to mobility, appearance and disappearance of nodes. The key operation of the algorithm is each node's 1-hop broadcast of its own estimated trajectory and update of its own trajectory on receiving the trajectory information from its neighbors. Although there are some methods that utilize the trajectory information to decrease the error of estimated positions [2, 4], they are designed as centralized algorithms. Unlike these algorithms, we need to identify the "truly effective information" for each mobile node in estimating its position to efficiently use limited bandwidth. Also we have compared our algorithm with the existing on-line algorithms of estimating mobile nodes' positions [6, 14] to show the advantage of using trajectory information. (2) Regarding the above, we try to determine how much information about trajectories should be exchanged among nodes based on the analysis of algorithm property and performance analysis by simulations. As far as we know, there is no existing work on decentralized trajectory estimation. (3) We give two realistic applications to demonstrate the applicability and effectiveness of our protocol in the real world.

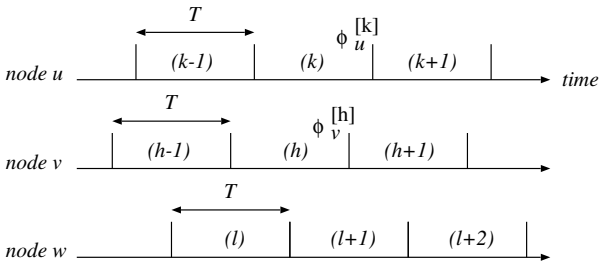


Figure 1: Time intervals of nodes.

3. TRADE OVERVIEW

3.1 Environment and Assumption

The TRADE algorithm is executed on each mobile terminal called *mobile node* (or simply *node*), which is equipped with a PAN communication device such as IEEE802.15.4. We assume another type of nodes called *landmarks*, which can know their exact positions anytime. A landmark may be a mobile node with an accurate GPS receiver or may be a base station. Each mobile node or landmark is assumed to have a unique ID that can be generated from its MAC address or a random value.

Since our method is a range-free localization, each node does not have to measure the distance between nodes. We only assume the maximum radio range and the maximum speed of all the nodes. They are denoted by R and V, respectively.

We let each node have an integer timer that is incremented for every time interval T. This common time interval T is used by all the nodes. Hereafter, node u's k-th time slot is denoted by $\phi_u^{[k]}$. We do not assume that these timers are synchronized (Fig. 1). We let $\phi_u^{[k]} - \phi_v^{[h]}$ denote the number of time slots by which $\phi_u^{[k]}$ precedes $\phi_v^{[h]}$. For example, in Fig. 1, $\phi_u^{[k]} - \phi_v^{[h]} = 1$, $\phi_u^{[k]} - \phi_v^{[h+1]} = -1$ and $\phi_u^{[k+1]} - \phi_u^{[k-1]} = 2$.

3.2 Basic Operation

Table 1: Notations.

Symbols	Descriptions
T	unit of time
V	maximum speed of nodes
$\phi_u^{[k]}$	node u's k-th time slot
$p_u^{[k]}$	estimated position of node u in $\phi_u^{[k]}$
$P_u^{[k,k']}$	estimated trajectory of node u between $\phi_u^{[k]}$ and $\phi_u^{[k']}$ (*i.e.,* $p_u^{[k]}, p_u^{[k+1]}, \ldots, p_u^{[k']}$)
$F^{(1)}$	1-hop forwarding interval, *i.e.,* the number of time slots to send its own trajectory information as a hello message
$F^{(2)}$	2-hop forwarding interval, *i.e.,* the number of time slots to include 1-hop neighbors' trajectory information into a hello message
W	the length of trajectory used to predict the current position
L	the length of trajectory each node updates

Hereafter, $p_u^{[k]}$ denotes the representative estimated position of node u in time slot $\phi_u^{[k]}$. Also we let $P_u^{[k,k']}$ denote the estimated trajectory of node u between time slots $\phi_u^{[k]}$ and $\phi_u^{[k']}$ ($k \leq k'$). We represent the trajectory $P_u^{[k,k']}$ by the sequence of the estimated positions, $p_u^{[k]}, p_u^{[k+1]}, \ldots, p_u^{[k']}$. The notations are summarized in Table 1.

Message Exchange.

In our proposed TRADE algorithm, each node u periodically broadcasts its own estimated trajectory as a *hello message* once every $F^{(1)}$ time slots. $F^{(1)}$ is called *1-hop forwarding interval* that indicates how often each node forwards its own estimated trajectory. Furthermore, for every $F^{(2)}$ time slots, node u may include its 1-hop neighbors' estimated trajectories received from those neighbors. $F^{(2)}$ is called *2-hop forwarding interval* that indicates how often each node forwards the estimated trajectories received through hello messages. $F^{(1)}$ and $F^{(2)}$ are positive integers, and $F^{(2)}$ is an integer multiple of $F^{(1)}$. To save traffic, a large value may be set to $F^{(2)}$. Consequently, each node knows its 1- or 2-hop neighbors' estimated trajectories.

In more details, a hello message from node u in $\phi_u^{[k]}$ contains node u's estimated trajectory $P_u^{[k',k]}$ ($k' = k - L + 1$) where L is a positive integer called *the length of trajectory*. For every $F^{(2)}/F^{(1)}$ hello messages, node u may also include node v's estimated trajectory $P_v^{[h',h]}$ ($\phi_u^{[k']} - \phi_v^{[h']} \leq 1$, $\phi_u^{[k'']} - \phi_v^{[h]} \leq 1$ and $k' \leq k'' < k$) where $\phi_u^{[k'']}$ is the latest time slot when v is the 1-hop neighbor of node u. If node u receives a hello message from node v in time slot $\phi_u^{[k]}$, it knows the estimated trajectory $P_v^{[h',h]}$ ($h' = h - L + 1$) of 1-hop neighbor v and the estimated trajectory $P_w^{[l',l]}$ of each 2-hop neighbor w (if any). We note that each landmark includes its accurate position information into its hello messages.

Position Estimation.

We denote the current time slot of node u by $\phi_u^{[\kappa]}$. In every time slot, node u predicts current position $p_u^{[\kappa]}$ from its last W estimated positions $P_u^{[\kappa-W,\kappa-1]}$ where W is a positive integer. The estimation of $p_u^{[\kappa]}$ is done by taking the average sum of vectors in trajectory $P_u^{[\kappa-W,\kappa-1]}$ per a time slot for the last W time slots. That is,

$$p_u^{[\kappa]} = p_u^{[\kappa-1]} + \frac{1}{W}\sum_{k=1}^{W}\left(p_u^{[\kappa-k]} - p_u^{[\kappa-(k+1)]}\right) \quad (1)$$

Trajectory Update.

After the prediction of the current position, node u updates its own last L estimated positions $P_u^{[\kappa-(L-1),\kappa]}$ where L is a constant using (i) the estimated trajectories of 1-hop (and 2-hop) neighbors and (ii) the *history of neighbors* that indicates the 1-hop or 2-hop neighbors in each past time slot.

To update the last L estimated positions, we consider the following three types of relationship between the two nodes' positions at the same time or between the two positions of a single node at previous and next time slots. They are called (i) *positive radio-range* constraint, (ii) *negative radio-range* constraint and (iii) *movable-range* constraint, and are defined as follows.

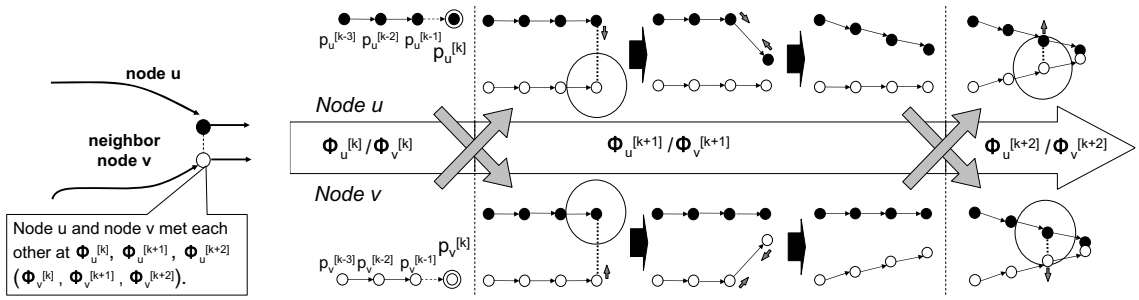
1. For any time slot $\phi_u^{[k]}$ and any node v, if node v was a 1-hop neighbor of node u in time slot $\phi_u^{[k]}$ and if the distance between $p_u^{[k]}$ and $p_v^{[h]}$, where $\phi_u^{[k]} - \phi_v^{[h]} = 1$, is larger than R, node u should update $p_u^{[k]}$ such that the distance is shorter than R. This is a positive radio-range constraint.

2. For any time slot $\phi_u^{[k]}$ and any node v, if node v was *NOT* a 1-hop neighbor of node u in time slot $\phi_u^{[k]}$ and if the distance between $p_u^{[k]}$ and $p_v^{[h]}$, where $\phi_u^{[k]} - \phi_v^{[h]} = 1$, is shorter than R, node u updates $p_u^{[k]}$ such that the distance is larger than R. This is a negative radio-range constraint.

3. For any time slot $\phi_u^{[k]}$, if the distance between $p_u^{[k]}$ and $p_u^{[k-1]}$ is larger than $V \cdot T$, node u updates $p_u^{[k]}$ such that the distance is shorter than $V \cdot T$. This is a movable-range constraint.

Because the positions of nodes may relate to each other temporally and spatially, update of one position may violate the constraints on the other positions in different time slots at the same node. Here, suppose that position $p_u^{[k]}$, which satisfies a movable-range constraint, is updated by a positive radio-range constraint. This may violate the constraints on positions $p_u^{[k-1]}$ and $p_u^{[k+1]}$. Therefore, we apply the update of trajectory $P_u^{[\kappa-(L-1),\kappa]}$ that is started from the earliest position $p_u^{[\kappa-(L-1)]}$ to the latest one $p_u^{[\kappa]}$ several times so that we can satisfy as many constraints as possible.

We exemplify this procedure in Fig. 2. For example, we suppose that nodes u and v met as shown in Fig. 2(a). Also, we show the estimation process of node u and v in Fig. 2(b). At first, node u predicts the current position $p_u^{[k]}$. In time slot $\phi_u^{[k+1]}$, node u modified $p_u^{[k]}$ to satisfy the positive radio-range constraint with $p_v^{[k]}$ based on the hello message sent from node v in time slot $\phi_u^{[k]}$. Since this modification violates some movable-range constraints, $P_u^{[k-2,k]}$ is modified to satisfy the movable-range constraints again. Moreover, $p_u^{[k-1]}$ is modified to satisfy the negative radio-range constraint with $p_v^{[k-1]}$ in time slot $\phi_u^{[k+2]}$ based on the hello message sent from node v in time slot $\phi_u^{[k+1]}$.



(a)Real movement (b)Estimation and modification of $P_u^{[k-3,k]}$ and $P_v^{[k-3,k]}$

Figure 2: Snapshot from algorithm.

In this way, the updated trajectory of node u may affect trajectories of neighbors by positive or negative radio-range constraints. Continuing the update of trajectory at local nodes and exchange of the updated trajectories, the trajectories of all the nodes are refined asymptotically.

The detail description of the algorithm is given in Appendix.

4. VALIDATION OF PROTOCOL DESIGN

4.1 Accuracy Improvement by Constraints

In this section, we explain how each constraint helps us to improve the accuracy of estimated trajectory through a typical example to validate our protocol design.

Assume that nodes a, b, c and d move as shown in Fig. 3(a). Here, the accurate position of node u in time slot $\phi_u^{[t]}$ is denoted as $q_u^{[t]}$. At time slot $\phi_a^{[3]}$, node b sends a hello message to a, node d sends a hello message to c, and node c sends a hello message to a. Hereafter, we explain how node a updates its current position and trajectory. For simplicity of discussion, we assume that neighbor nodes b, c and d have already estimated their trajectories with enough accuracy ($u \in \{b, c, d\} : p_u^{[t]} \simeq q_u^{[t]}$) and the almost accurate positions are recorded in the hello messages.

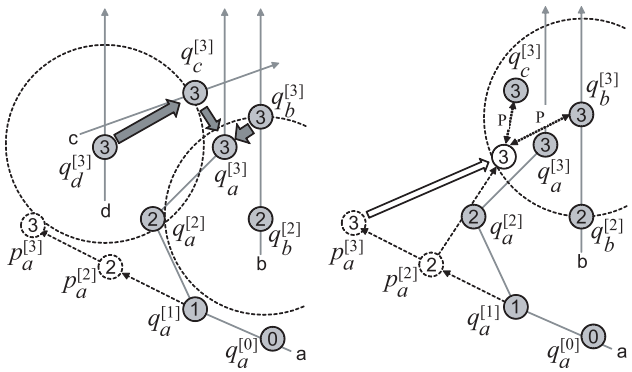
First, we explain that it helps us for accurate estimation to re-compute the trajectory using the positive radio-range constraints and movable-range constraints. In Fig. 3(b), we show the estimation result in the simplest way that estimates only the current position using the current positions of neighbor nodes (without trajectory information). In this case, node a receives the node b's hello message at $\phi_a^{[3]}$, but the estimated a's position $p_a^{[3]}$ is too far from the received b's position $p_b^{[3]}$. This means that position $p_a^{[3]}$ violates the positive radio-range constraint, and an error collection vector (see Appendix for definition) towards $p_b^{[3]}$, which is used to correct the position to satisfy the violated constraints, is applied to $p_a^{[3]}$. Repeating such position re-calculation, $p_a^{[3]}$ will be moved to an appropriate position.

On the other hand, Fig. 3(c) shows the estimation result when each node not only estimates the current position but also modifies the trajectory using trajectories of 1-hop neighbors. In this case, an error collection vector towards $p_b^{[3]}$ is also applied to $p_a^{[3]}$. Additionally, appropriateness of a's trajectory is considered. The movable-range constraints, which restrict the distances between $p_a^{[1]}$ and $p_a^{[2]}$ and between $p_a^{[2]}$ and $p_a^{[3]}$, are applied, and the positions $p_a^{[1]}$ and $p_a^{[2]}$ are also updated.

Next, we explain the effect of negative radio-range constraints on estimation of the trajectories. We consider that node a can receive d's position $p_d^{[3]}$ forwarded by c. The estimation result by using the negative radio-range constraints from 1-hop and 2-hop neighbors is shown in Fig. 3(d). In this case, since a did not receive any message from b around $\phi_a^{[2]}$, they are considered as having been apart from each other. So an error collection vector to the opposite direction from $p_b^{[2]}$ is applied to $p_a^{[2]}$. Similarly, a did not receive any message from d around $\phi_a^{[3]}$, a's position $p_a^{[3]}$ will be excluded from the radio-range of d. As the result, more accurate trajectory can be estimated than in the case of Fig. 3(c). Since the estimated a's current position $p_a^{[3]}$ still includes relatively larger error, $p_a^{[3]}$ will be updated after $\phi_a^{[4]}$ repeatedly whenever a receives a hello message. This will make the estimation more accurate.

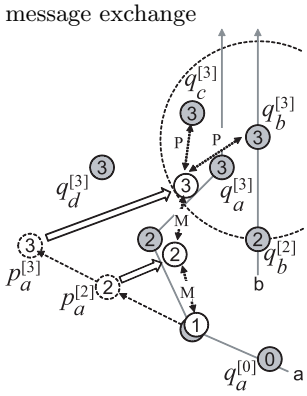
However, we note that because a negative radio-range constraint makes positions of nodes pushed away from a circle, it may obstruct desirable update if the initial estimation is too far from the accurate one. For example, in the situation shown in Fig. 3(e), a negative radio-range constraint disturbs convergence of the solution. Here, node a had gone straight and turned at $\phi_a^{[1]}$. Then a encounters with b at $\phi_a^{[3]}$, and receives a hello message. Node a estimates $p_a^{[2]}$ and $p_a^{[3]}$ by linear interpolation. After receiving a hello message, $p_a^{[3]}$ is moved toward $p_b^{[3]}$ by a positive radio-range constraint. However, around $\phi_a^{[2]}$, a did not receive any hello message, since a was not in the radio-range of b. So a negative radio-range constraint is applied to the pair of $p_a^{[2]}$ and $p_b^{[2]}$. As shown in the figure, the constraint prohibits $p_a^{[2]}$ from moving to $p_b^{[2]}$, and $p_a^{[2]}$ may not be converged to an accurate position. Such a problem occurs only if forbidden areas made by negative radio-range constraints lie in the area between the initial positions and the accurate positions. Hence, this problem can be avoided by applying negative radio-range constraints only for the positions that are relatively near from the accurate ones. We thought we could avoid this problem by skipping application of negative radio-range constraints to the recent positions, which may include large error. This is proved by simulations in Sec. 5.1.

4.2 Demonstration of TRADE by Simulations

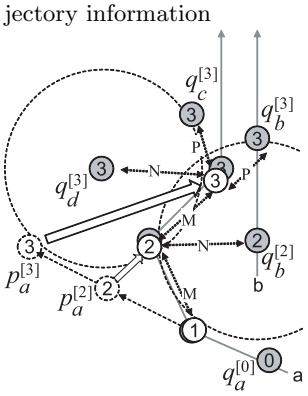


(a) Real movement and message exchange

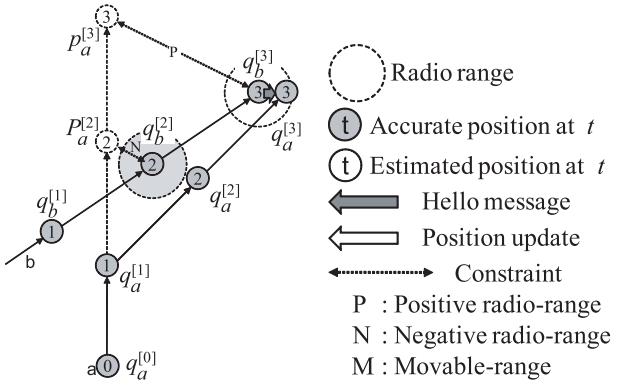
(b) Estimation without trajectory information



(c) Estimation without negative radio-range constraints



(d) Estimation with negative radio-range constraints



(e) Bad influence of negative radio-range constraints on trajectory update

Figure 3: Overview of estimation process.

To prove the property analysis in Sec. 4.1, we have conducted simulations of TRADE.

We assume the Random WayPoint (RWP) mobility model in a 100m × 100m area with 9 landmarks (denoted by black stars in Fig.4). The simulation time S was set to 300 sec. The speeds of nodes were set to follow the uniform distribution in the interval $[1.0, 2.0]$ m/s. The pause time of RWP was set to zero. In order to see the nature of the algorithm, we would like to exclude the effect of wireless communication instability. Therefore, in this experiment, we have used the ideal communication model where two nodes can communicate with each other without loss of data if the distance

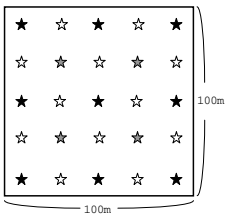


Figure 4: Landmark deployment patterns.

between them is not greater than R. The default values of the other parameters are as follows. The interval T of time slot was 1.0 sec., the length L of a trajectory modified in each update was 10, the length W of a trajectory used to predict the current position $p_u^{[k]}$ was 10, radio range R was 10 m, the number n of nodes was 200, the 1-hop forwarding interval $F^{(1)}$ was 1 sec., and the 2-hop forwarding interval $F^{(2)}$ was ∞. We note that $F^{(2)} = \infty$ means that nodes never sent the trajectories of their 1-hop neighbors.

We have measured the position errors defined by formulas (2) and (3), where $p_u^{[k][k']}$ is the value of $p_u^{[k]}$ as of time slot $\phi_u^{[k']}(k \leq k')$, and $q_u^{[k]}$ is the accurate position of node u in time slot $\phi_u^{[k]}$. Also, l is called *elapsed time* from k.

$$Error(k,l) = \frac{1}{n}\sum_{u=1}^{n} \| q_u^{[k]} - p_u^{[k][k+l]} \| \tag{2}$$

$$Error(l) = \frac{1}{S/T}\sum_{k=1}^{S/T} Error(k,l) \tag{3}$$

Intuitively, $Error(k,l)$ represents the average error of all the nodes' estimated positions of k-th time slot which are referred in $(k+l)$-th time slot. $Error(l)$ is the average of $Error(k,l)$ throughout the simulation. Hereafter, $Error(0)$ is called *realtime error* and $Error(L-1)$ is called *final error* since no position update is applied to $p_u^{[k]}$ if $L \leq l$.

First, we show that the exchange of trajectory information is effective to improve position accuracy since it provides negative radio-range constraints. According to the TRADE algorithm, node u predicts the current position $p_u^{[k]}$ in k-th time slot $\phi_u^{[k]}$ and updates it in the following $L-1$ time slots (*i.e.*, $\phi_u^{[k+1]}$, $\phi_u^{[k+2]}$, ..., $\phi_u^{[k+L-1]}$). Fig. 5 shows $Error(l)$ $(0 \leq l < L$ and $L = 10)$ and the number of positive or negative radio-range constraints used for position updates in time slots $\phi_u^{[k+1]}$, $\phi_u^{[k+2]}$, ..., $\phi_u^{[k+L-1]}$ $(0 \leq k \leq S/T - L)$. We can observe that the position error is decreasing as time elapses because node u may obtain new negative radio-range constraints whenever it receives a hello message.

Secondly, we show that the 1-hop neighbors' trajectory information gradually increases negative radio-range constraints. We also show that it provides almost the same number of constraints as the 2-hop neighbors' position information but improves estimation accuracy. We conducted experiments with different settings of length L and 2-hop forwarding interval $F^{(2)}$, in order to see how and when negative radio-range constraints are delivered to each node. For estimation of $p_u^{[k]}$ in the case of $L = 2$ and $F^{(2)} = 1$, the number of positive radio-range constraints of $\phi_u^{[k]}$ was 8.55, and that of negative radio-range constraints was 9.16. We had almost the same number of constraints as of $\phi_u^{[k+L-1]}$

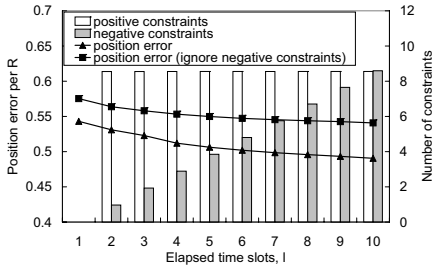


Figure 5: Number of positive / negative constraints and position error (vs. elapsed time slots l). $L = 10$, $F^{(1)} = 1$ and $F^{(2)} = \infty$.

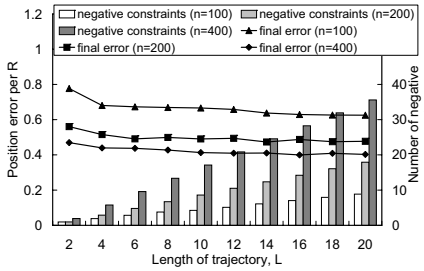


Figure 6: Number of negative constraints and final error (vs. trajectory length L). $F^{(1)} = 1$ and $F^{(2)} = \infty$.

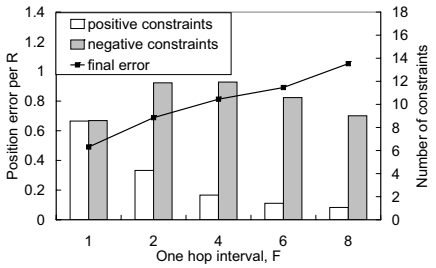


Figure 7: Number of positive / negative constraints and final error (vs. 1-hop forwarding interval $F^{(1)}$). $L = 10$ and $F^{(2)} = \infty$.

in the case of $L = 10$ and $F^{(2)} = \infty$. This indicates that if each hello message does not contain any 1-hop neighbor's trajectory, a similar number of constraints can be obtained through nodes' own trajectory information with some delay. Although they had almost the same number of constraints, the resulting final errors were different. In the case of $L = 2$ and $F^{(2)} = 1$, $Error(L-1) = 0.81R$ while in the case of $L = 10$ and $F^{(2)} = \infty$, $Error(L-1) = 0.50R$. The difference between the two cases is the timing to obtain the negative radio-range constraints; in the former case they are delivered almost in real-time and in the latter case they are delivered with delays. This fact proves occurrence of the case of Fig. 3(e) in Sec. 4.1 and indicates that application of negative radio-range constraints should be delayed. In TRADE, the positive radio-range constraints can be delivered in real-time and the negative radio-range constraints are gradually derived in the subsequent time slots by exchange of nodes' trajectory information.

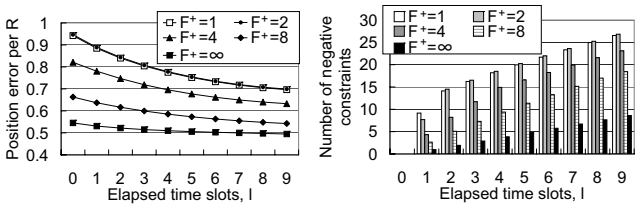
5. PERFORMANCE ANALYSIS

In this section, we discuss several factors that affect the accuracy based on simulation results. We also analyze the traffic amount of TRADE to see its appropriateness. We assume the same simulation settings as in the previous section, and use the default parameters if not otherwise specified.

5.1 Estimation Accuracy

Trajectory Length: We observe the effect of L on the final errors. Fig. 6 shows the final errors and the number of negative radio-range constraints used in time slot $\phi_u^{[k+L-1]}$. To exclude the effect of the number of neighboring nodes on the analysis, we conducted simulations with different settings of the number n of nodes, n =100, 200 and 400. From the graph, we can say that larger L makes better accuracy in the final errors, because longer trajectories bring more negative radio-range constraints as proved in the previous section. It is worth noting that the average numbers of positive radio-range constraints were 4.27, 8.55 and 16.99 in the cases of $n = 100$, 200 and 400, respectively. We may choose $L = 10$ to obtain the similar number of negative constraints with that of positive ones, which may result in better accuracy.

1-hop Forwarding Interval: Fig. 7 shows the final errors for the different values of 1-hop forwarding interval, $F^{(1)}$. From the result, larger 1-hop forwarding interval makes larger error because the number of positive radio-range constraints decreases. On the contrary, the numbers of negative radio-range constraints is maximum with $F^{(1)} = 2$ and $F^{(1)} = 4$.



(a) Position error (b) Number of negative constraints

Figure 8: Number of negative constraints and position error (vs. 2-hop forwarding interval $F^{(2)}$). $L = 10$ and $F^{(1)} = 1$.

The reason is as follows. With $F^{(1)} = 6$ or more, each node may miss to receive hello messages from its neighboring nodes considering the sojourn time in the radio circle, and the number of both positive and negative radio-range constraints may decrease. With $F^{(1)} = 2$ or 4, each node may not miss hello messages. But the number of positive radio-range constraints is less than $F^{(1)} = 1$. Instead, the number of negative radio-range constraints increases because of the decrease of positive radio-range constraints.

2-hop Forwarding Interval: In the TRADE algorithm, each node u sends 1-hop neighbors' estimated trajectories for every $F^{(2)}$ time slots. We show $Error(l)$ $(0 \le l < L)$ in Fig. 8(a) under various values of $F^{(2)}$. Also, the number of negative radio-range constraints used for estimation is shown in Fig. 8(b). The number of positive radio-range constraints is 8.55. From the result, smaller $F^{(2)}$ makes larger final error even with more negative radio-range constraints. Due to the nature of the algorithm that iteratively corrects positions, it may not be possible to satisfy all the constraints in early time slots if we have too many ones. By this feature, errors in the early slots may be propagated to the initial positions in the next time slot. In order to prove this feature, we show in Fig. 9 the position errors of TRADE which does not use negative radio-range constraints in a certain time slots from realtime. From the result, we can see that the realtime and final errors are improved if we do not use negative radio-range constraints in the early slots, especially between 0 and 2 time slots. However, we do not use the negative radio-range constraints for more than two time slots, the final errors get worse because of insufficiency of constraints. From the result, in the following simulations, we evaluate TRADE which ignores negative radio-range constraints in the first two slots.

Node Density and Radio Range: We have evaluated the impact of the node density with different values of the num-

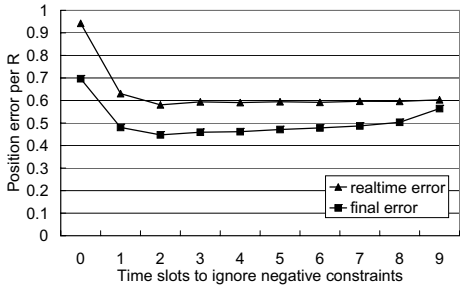


Figure 9: Position error (vs. time slots to ignore negative constraints). $L = 10$, $F^{(1)} = 1$ **and** $F^{(2)} = 1$

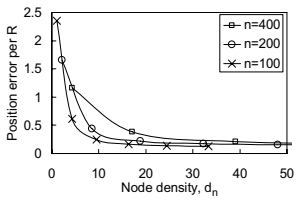


Figure 10: Final error (vs. node density).

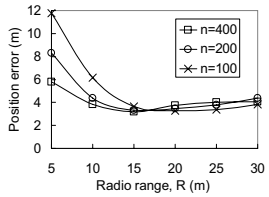


Figure 11: Final error (vs. radio range).

ber n of nodes and radio range R. We let d_n denote the node density, *i.e.,* the average number of neighboring nodes. At first, Fig.10 shows the final errors for various values of radio range R and node density d_n. From the result, higher node density achieves more accurate estimation. Especially, the final errors are less than $0.2R$ where d_n is more than 20 for all the cases of n. Also, smaller number of nodes n makes better accuracy in the same node density. This is because we adjusted the radio ranges to realize the same node density with smaller number of nodes, and it causes many chances to communicate with landmarks. Secondly, Fig. 11 shows the final errors for various values of n. From the result, when the radio range is less than about 15m, larger radio ranges decrease errors because it also provides more chances to communicate with landmarks. However, when the radio range is more than 20m, it often makes errors larger since radio-range constraints are relaxed.

5.2 Communication Traffic

We analyze the amount of communication traffic of TRADE. We can represent a piece of position information by 8 bytes (= 4 bytes $\times 2$). So the message size is $8L$ bytes where L is the trajectory length to be transmitted, and the size of a message with 1-hop neighbors' trajectory information sent for every $F^{(2)}/F^{(1)}$ hello message is about $8L+8Ld_n$, where d_n denotes the node density. So, the average message size is $8L + (8Ld_nF^{(1)}/F^{(2)})$. In the case of $n = 200$, $L = 10$, $F^{(1)} = 1$, $F^{(2)} = 4$ and $R = 10$, where d_n is about 8.54, the average message size is 250 bytes. In this situation, the consumed bandwidth by each node is about 2 kbps. This is small enough considering the recent PAN technologies like IEEE802.15.4, and even we may control it by adjusting the length L and interval $F^{(2)}$.

6. PERFORMANCE EVALUATION

In this section, we have conducted four types of simulations to evaluate the performance of TRADE. In the first simulation, we have evaluated the impact of the factors that

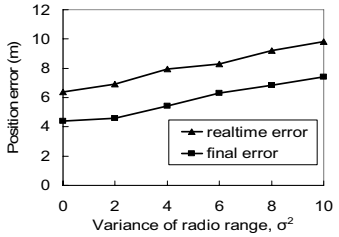


Figure 12: Position error (vs. variance of radio range).

seem to affect TRADE in real environments. In the previous section, we analyze the estimation accuracy of TRADE in ideal environments. In real environments, however, several factors such as packet loss and irregularity of radio range affect the estimation accuracy. In the second simulation, we have compared TRADE with the existing real-time localization methods. In this environment, we show that TRADE can achieve more accurate estimation than the existing methods by exchange of trajectories. In the third simulation, we consider the issues on service deployment of TRADE such as deployment of landmarks and member dynamics. In the fourth experiments, we demonstrate the performance of TRADE with two application scenarios. We assume the same simulation settings as in the previous section except in the fourth simulation. Also, we use the default values except $F^{(2)}$ ($F^{(2)} = 4$) if not otherwise specified.

6.1 Applicability to Real Environments

Irregularity of Radio Range: In real environment, the radio range may become irregular because of interference, reflections and other reasons. To evaluate the impact of irregular radio range, we assume that the radio range follows the normal distribution denoted by $R = N(10, \sigma^2)$m. Now, Fig. 12 shows the realtime errors and final errors with various values of variance σ^2 of the radio range. From the result, larger variance of the radio range makes larger errors because the radio-range constraints become more inaccurate.

Packet Loss: In the previous simulations, we did not consider message loss. As previously discussed, we show that higher node density makes better estimation, but hello messages of large sizes may be lost because of their contention on the same wireless channel. In order to see the effect by message contention, we used the following contention model $P = (1-p)^{d_n}$ where P is the probability that a node succeeds to receive a packet from its neighbors, d_n is node density, and p is the occupancy ratio of the channel, *i.e.,* the average sending data size/channel capacity. We think that the channel capacity is 250kbps assuming PAN bandwidth like IEEE802.15.4. Table 2 shows the final errors under this model. From the result, although larger number of nodes makes packet loss rates worse, it has less impact on the final errors. We can say that considering the wireless channel capacity limitation, TRADE can achieve reasonable estimation accuracy.

Evaluation with Real Environment Communication Log: In order to examine the performance in a real environment, we have used communication log among MICAz Motes terminals collected in the experiments of our previous work[4]. In the experiment, we prepared an area of 18m $\times$ 18m consisted of 9m $\times$ 9m cells, and let 10 students having Motes

Table 2: Position error (with channel contention model).

n	100	200	300
Packet loss rate	0.01	0.05	0.17
Final error with contention model	0.62m	0.46m	0.37m
Final error without contention model	0.60m	0.46m	0.38m

Table 3: Results of real environment experiments.

	Realtime error	Final error
Real environment	3.46m	3.14m
Simulation (R =3.0 m)	4.10m	4.02m
Simulation (R =3.8 m)	2.90m	2.19m

walk on the edges of the cells choosing their directions at each crossing point. We put 4 Motes as landmarks at the corners of the area. The radio transmission power of Mote was set to -15dBm, and radio transmission range R was about 3.0m. The experimental result in comparison with simulation is shown in Table 3. We can see that the estimation error in the simulation with $R = 3.0$ is larger than that in the real environment. This is because in the real environment, actual transmission ranges were longer than expected ones due to irregularity of radio ranges. Actually, with $R = 3.8$ where the average number of neighboring nodes is the same as that of the real connectivity information, the difference between the results of the real and simulation environments becomes smaller. We could confirm the similarity between the experiments by simulation and in the real environment, and the good aspect of TRADE proved by simulation can be retained in the real environment.

6.2 Comparison with Other Algorithms

We have compared the performance of TRADE with that of Amorphous[11] and UPL[14]. We regard not final errors but realtime errors as the estimation errors in TRADE for fair comparison. The result is shown in Fig. 13. Especially in the case of a small number of nodes, the accuracy of TRADE is better than the others because of exchange of trajectory.

6.3 Other Issues in Real Environment

In real environments, nodes may join the TRADE protocol in different timings. When a node starts the algorithm, it has no information about its position and needs a certain time to estimate positions with reasonable errors. We call this time *unstable time*. In order to evaluate unstable time, we let 100 new nodes start the algorithm at the half time of the simulation when 200 existing nodes had already started the algorithm. The initial deployment of new nodes was uniform distribution. Fig.14 shows the position errors of the existing nodes and new nodes over time where at time 0 new nodes started the algorithm. From the result, the errors of new nodes were almost the same as those of the existing ones after 15 seconds. Also, the position errors of the existing nodes got worse when new nodes joined, but they were decreasing rapidly.

Furthermore, in order to see the behavior of the protocol in different environments, we have evaluated the impact of landmark density in the following cases: (1) 9 landmarks are deployed at black stars in Fig.4 (default), (2) 13 landmarks are deployed at black and gray stars in Fig.4, (3) landmarks are deployed at black and white stars in Fig.4 and (4) landmarks are deployed at all the stars in Fig.4. Fig.15 shows the realtime and final errors in all the cases. From the result, more landmarks can achieve smaller realtime and final errors.

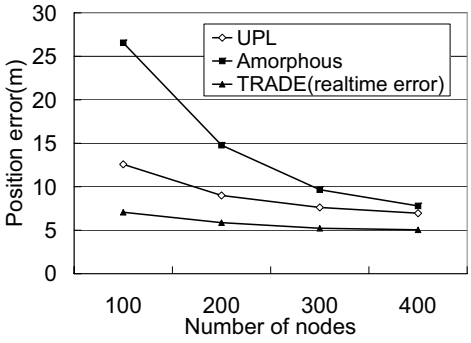


Figure 13: Comparison with other algorithms.

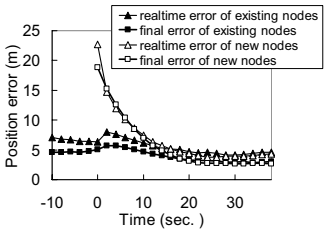


Figure 14: Position error(vs. unstable time).

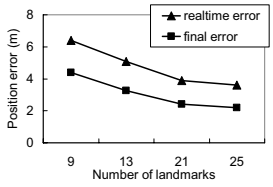


Figure 15: Position error(vs. landmark density).

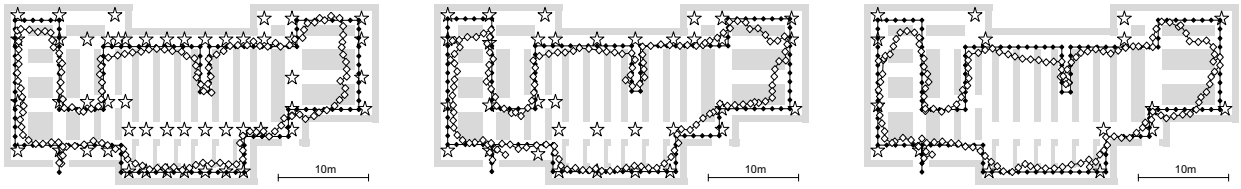
6.4 Application Examples

We show two application scenarios, and demonstrate that our TRADE works fine by simulation experiments.

Customer Tracking and Location-aware Shopping Support.

In huge stores and shopping malls, by attaching small sensor nodes to shopping carts, we can track the behavior of customers by collecting their trajectories. Such information can be used to provide highly-personalized location-aware service and advertisement by predicting their behavior, objective and interest in stores. Also, customers can make use of their trajectory information to understand the visited and current locations and to determine shopping plans for the remaining time. To evaluate the performance of TRADE in this environment, we have conducted simulations using a store map shown in Fig. 16, and have estimated the trajectory of a customer from the entrance to the exit. We put other 40 nodes moving in the store randomly along the pathways. We have prepared three different patterns of landmark placements; (a) the dense placement in which 53 landmarks are placed at all the corners, (b) the medium placement in which 35 landmarks are placed at major corners, and (c) the least placement in which only 16 landmarks are placed on the walls. We have used a radio model where the radio range follows a normal distribution $N(5.0, 0.5)$m.

In Fig. 16, we show the estimated trajectories (lines with



(a) dense placement (b) medium placement (c) least placement

Figure 16: Estimated and real trajectories in store with different placement of landmarks.

Table 4: Final errors in the store example.

Number of landmarks	Average (m)	Worst (m)
53	0.78	2.43
35	0.86	2.81
16	0.94	3.40

white circles) and the real trajectories (lines with black circles) with landmarks (stars). We also show the estimation errors in Table 4. From these results, even with the least placement of landmarks, the trajectory could be estimated clearly and accurately with less than 1m error in average, and its shape is very close to the real trajectory. In particular, even though the real trajectory goes into the pathway between shelves and goes back and around the place where there is no landmark, TRADE could estimate such complex behavior. This shows that TRADE can be applied to such an application that needs to identify the location of indoor nodes precisely with a reasonable number of landmarks.

Group Navigation in Amusement Park.

As the second example, we consider crowded outdoor environment where many people walk around a place like an amusement park. Since TRADE allows nodes to obtain neighbors' trajectories, we may provide such a service which warns parents that their children are getting separated from them. We use a map shown in Fig. 17 constructed by referring to a Google satellite picture. We assume that mobile nodes may appear and disappear at specific points such as entrance, shops and attractions. The average number of nodes at each moment was 606. We also put two special nodes as a pair of guests playing together. They move around the park getting closer and further. From Fig. 18, even though certain difference from the real distance is observed around 200 seconds, we can say that the estimation could reproduce the variation of the distance between the two nodes. This shows that TRADE can estimate the relative positions accurately.

7. CONCLUSION

In this paper, we have proposed a novel trajectory estimation method called TRADE in fully decentralized mobile ad hoc networks. In TRADE, each mobile node periodically transmits messages containing its estimated trajectory information, and re-computes its own trajectory using those from its neighbors. Our protocol design has been validated by property analysis and a number of simulation results. Furthermore, we have shown the effectiveness of the protocol in the real world using two realistic application examples.

Our future work is to evaluate the usefulness of TRADE under more scalable and practical applications.

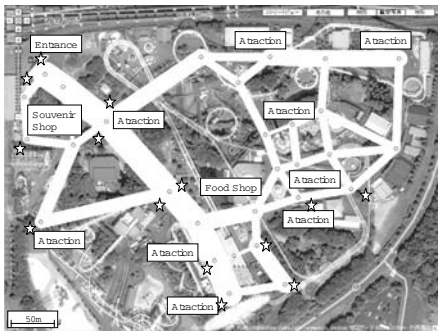


Figure 17: Amusement park map.

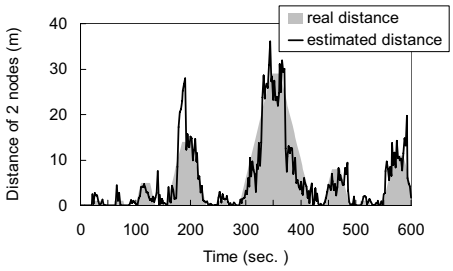


Figure 18: Distance between two nodes.

8. REFERENCES

[1] P. Bahl and V. N. Padmanabhan. RADAR: an in-building RF-based user location and tracking system. In *Proc. of INFOCOM 2000*, pages 775–784, 2000.

[2] J. M. Cabero, F. D. la Torre, A. Sanchez, and I. Arizaga. Indoor people tracking based on dynamic weighted multidimensional scaling. In *Proc. of MSWiM 2007*, pages 328–335, 2007.

[3] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua. Multicamera people tracking with a probabilistic occupancy map. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):267–282, 2008.

[4] S. Fujii, A. Uchiyama, T. Umedu, H. Yamaguchi, and T. Higashino. An off-line algorithm to estimate trajectories of mobile nodes using ad-hoc communication. In *Proc. of PerCom 2008*, pages 117–124, 2008.

[5] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for

large scale sensor networks. In *Proc. of MobiCom 2003*, pages 81–95, 2003.

[6] L. Hu and D. Evans. Localization for mobile sensor networks. In *Proc. of MobiCom 2004*, pages 45–57, 2004.

[7] A. Kleiner and D. Sun. Decentralized SLAM for pedestrians without direct communication. *Proc. of International Conference on Intelligent Robots and Systems (IROS 2007)*, pages 1461–1466, 2007.

[8] H. Lee, M. Wicke, B. Kusy, and L. Guibas. Localization of mobile users using trajectory matching. In *Proc. of MELT 2008*, pages 123–128, 2008.

[9] M. Li and Y. Liu. Rendered path: range-free localization in anisotropic sensor networks with holes. In *Proc. of MobiCom 2007*, pages 51–62, 2007.

[10] B. T. Morris and M. M. Trivedi. A survey of vision-based trajectory learning and analysis for surveillance. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(8):1114–1127, 2008.

[11] R. Nagpal, H. Shrobe, and J. Bachrach. Organizing a global coordinate system from local information on an ad hoc sensor network. In *Proc. of IPSN 2003*, pages 333–348, 2003.

[12] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket location-support system. In *Proc. of MobiCom 2000*, pages 32–43, 2000.

[13] Y. Shang, W. Rml, Y. Zhang, and M. Fromherz. Localization from connectivity in sensor networks. *IEEE Transaction on Parallel and Distributed Systems*, 15(11):961–974, 2004.

[14] A. Uchiyama, S. Fujii, K. Maeda, T. Umedu, H. Yamaguchi, and T. Higashino. Ad-hoc localization in urban district. In *Proc. of INFOCOM 2007 Mini-Symposium*, pages 2306–2310, 2007.

APPENDIX

A. DETAILS OF TRAJECTORY UPDATE

Node u applies the following algorithm to update the trajectory $P_u^{[\kappa-(L-1),\kappa]}$. To update the trajectory, for each of the three constraints (positive radio-range, negative radio-range and movable-range constrains), we provide the corresponding *error correction (EC) vector* and an *error correction (EC) coefficient*. The EC vector moves the current estimation to the direction so that the corresponding constraints can be satisfied, and the associated EC coefficient is used to balance the different EC vectors that are applied at a time. Then for each estimated position $p_u^{[k]}$ in $P_u^{[\kappa-(L-1),\kappa]}$, we check if it satisfies these constraints or not. If the current estimation $p_u^{[k]}$ violates some of the constraints, we obtain the average sum of the corresponding EC vectors weighted by EC coefficients for the new estimation.

Here, we formally describe the constraints that are applied to each estimated position $p_u^{[k]}$. (1) For each node v that was a neighbor of node u in time slot $\phi_u^{[k]}$, $\|p_u^{[k]} - p_v^{[h]}\| \leq R$, where $\phi_u^{[k]} - \phi_v^{[h]} \leq 1$, must hold based on the positive radio-range constraint. This constraint is referred to as $C^{pos}_{u,[k],v}$. (2) For each node v that was *NOT* a neighbor of node u in time slot $\phi_u^{[k]}$, $\|p_u^{[k]} - p_v^{[h]}\| > R$, where $\phi_u^{[k]} - \phi_v^{[h]} \leq 1$, must hold based on the negative radio-range constraint. This constraint is referred to as $C^{neg}_{u,[k],v}$. (3) For the estimated position of node u in time slot $\phi_u^{[k-1]}$, $\|p_u^{[k]} - p_u^{[k-1]}\| \leq V \cdot T$ must hold based on the movable-range constraint. This constraint is referred to as $C^{prev}_{u,[k]}$. Similarly, $C^{next}_{u,[k]}$ should also be satisfied for the estimated position $p_u^{[k+1]}$.

The EC vectors and EC coefficients are designed below. Hereafter, we let $\Delta p(C)$ and $w(C)$ denote the EC vector and the EC coefficient of constraint C.

For the positive radio-range constraint $C^{pos}_{u,[k],v}$, we give the following EC vector and EC coefficient. The EC vector is the vector between the two points. Therefore, as larger than R the distance is, the norm of the EC vector is larger.

$$\begin{aligned} \Delta p(C^{pos}_{u,[k],v}) &= p_u^{[k]} - p_v^{[h]} \\ w(C^{pos}_{u,[k],v}) &= \begin{cases} 1 & \text{if } C^{pos}_{u,[k],v} \text{ is violated} \\ 0 & otherwise \end{cases} \end{aligned} \quad (4)$$

Similarly, for the negative radio-range constraint $C^{neg}_{u,[k],v}$,

$$\begin{aligned} \Delta p(C^{neg}_{u,[k],v}) &= -(R - \|p_u^{[k]} - p_v^{[h]}\|) \cdot (\widehat{p_u^{[k]} - p_v^{[h]}}) \\ w(C^{neg}_{u,[k],v}) &= \begin{cases} 1 & \text{if } C^{neg}_{u,[k],v} \text{ is violated} \\ 0 & otherwise \end{cases} \end{aligned} \quad (5)$$

where $\widehat{a}$ is a unit vector of a.

For the movable-range constraint $C^{prev}_{u,[k]}$,

$$\begin{aligned} \Delta p(C^{prev}_{u,[k]}) &= p_u^{[k]} - p_u^{[k-1]} \\ w(C^{prev}_{u,[k]}) &= \begin{cases} 1 + \sum_v \{w(C^{pos}_{u,[h],v}) + w(C^{neg}_{u,[h],v})\} \\ \quad \text{if } C^{prev}_{u,[k]} \text{ is violated} \\ 0 \\ \quad otherwise \end{cases} \end{aligned} \quad (6)$$

The same EC vector and the EC coefficient are applied to the movable-range constraint $C^{next}_{u,[k]}$;

$$\begin{aligned} \Delta p(C^{next}_{u,[k]}) &= p_u^{[k]} - p_u^{[k+1]} \\ w(C^{next}_{u,[k]}) &= \begin{cases} w(C^{prev}_{u,[k]}) & \text{if } C^{next}_{u,[k]} \text{ is violated} \\ 0 & otherwise \end{cases} \end{aligned} \quad (7)$$

For each violated constraint, we add the corresponding EC vector weighted by its EC coefficient;

$$\begin{aligned} \Delta p_u^{[k]} &= \sum_v w(C^{pos}_{u,[h],v}) \Delta p(C^{pos}_{u,[k],v}) \\ &+ \sum_v w(C^{neg}_{u,[h],v}) \Delta p(C^{neg}_{u,[k],v}) \\ &+ w(C^{prev}_{u,[k]}) \Delta p(C^{prev}_{u,[k]}) + w(C^{next}_{u,[k]}) \Delta p(C^{next}_{u,[k]}) \end{aligned} \quad (8)$$

We also add the EC coefficients.

$$\begin{aligned} \Delta w_u^{[k]} &= \sum_v w(C^{pos}_{u,[h],v}) + \sum_v w(C^{neg}_{u,[h],v}) \\ &+ w(C^{prev}_{u,[k]}) + w(C^{next}_{u,[k]}) \end{aligned} \quad (9)$$

Finally, we update the estimated position $p_u^{[k]}$ as follows;

$$p_u^{[k]} \leftarrow p_u^{[k]} + \alpha \cdot \frac{\Delta p_u^{[k]}}{\Delta w_u^{[k]}} \quad (10)$$

where α is a constant to represent the degree of the effect of modifications. From several experiments, we have learned $\alpha = 0.3$ is almost the best value.

We continue to update each $p_u^{[k]}$ of trajectory $P_u^{[\kappa-(L-1),\kappa]}$ beginning with the oldest positions for Γ times. We may empirically choose $\Gamma = 10$.