RASS: A Real-time, Accurate and Scalable System for Tracking Transceiver-free Objects

Dian Zhang[1,2], Yunhuai Liu[4] and Lionel M. Ni[1,3]
Department of Computer Science and Engineering, Hong Kong University of Science and Technology[1]
College of Software, Shenzhen University[2]
Shanghai Key Lab of Scalable Computing and Systems, Shanghai Jiao Tong University[3]
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences[4]
Email:{zhangd@ust.hk, yunhuai@siat.ac.cn, ni@cse.ust.hk}

Abstract—**Transceiver-free object tracking is to trace a moving object without carrying any communication device in an environment where the environment is pre-deployed with some monitoring nodes. Among all the tracking technologies, RF-based technology is an emerging research field facing many challenges. Although we proposed the original idea, until now there is no method achieving scalability without sacrificing latency and accuracy. In this paper, we put forward a real-time tracking system RASS, which can achieve this goal and is promising in the applications like the safeguard system. Our basic idea is to divide the tracking field into different areas, with adjacent areas using different communication channels. So the interference among different areas can be prevented. For each area, three communicating nodes are deployed on the ceiling as a regular triangle to monitor this area. In each triangle area, we use a Support Vector Regression (SVR) model to locate the object. This model simulates the relationship between the signal dynamics caused by the object and the object position. It not only considers the ideal case of signal dynamics caused by the object, but also utilizes their irregular information. As a result it can reach the tracking accuracy to around $1m$ by just using three nodes in a triangle area with $4m$ in each side. The experiments show that the tracking latency of the proposed RASS system is bounded by only about $0.26s$. Our system scales well to a large deployment field without sacrificing the latency and accuracy.**

I. Introduction

Real-time tracking of moving objects based on the RF technology has been a hot research issue as it is important in many applications. Most of these applications require each object to carry a device as the transceiver to help estimate its location. However, such a requirement can hardly be met in some applications. For example, in a safeguard system, thieves will not carry any such device. Or even in some places, especially at home, where people do not always carry their tracking device (e.g., the cell phones) and they often leave them behind [1]. Therefore, low cost RF-based technologies to trace objects without carrying any device has attracted increasing research interests in recent years. It is called transceiver-free [3] [4] or device-free [2] object tracking.

To the best of our knowledge, we were the first group proposed the original idea [3], as well as provided some preliminary solutions and initial experimental results. Our first attempt was based on a wireless network composed of a number of communicating nodes (each node can be a simple sensor node without using its sensing ability). Nodes are periodically sending beacon messages. The system first detects the *Radio Signal Strength (RSS) dynamics* for each pair of communicating nodes, which is the difference of signal strength caused by the object. Then, it proposed a model of RSS dynamics to allow tracking transceiver-free objects. Based on this model, it utilized many such pairs of communicating nodes to locate the object.

However, the previous model is only based on one pair of nodes (one wireless link). Its accuracy may be improved by introducing more nodes to eliminate the noise behavior. But introducing more nodes will increase the possibility of interference among nodes. Moreover, each node has to wait for a backoff time in order to avoid beacon collision. The detection latency, therefore, will be dramatically increased. This problem becomes more serious in a large area deployment.

To solve the above problem, in this paper, we put forward a brand-new real-time tracking system RASS, which utilizes multiple communication channels to monitor different areas of the tracking field. In this way, we may prevent interference among communicating nodes belonging to different channels from happening. This is helpful to improve the tracking accuracy. More importantly, since we only need to consider the wireless communication among nodes in the same channel, the beacon interval of each node can be greatly shortened. As a result the tracking latency is dramatically decreased. Furthermore, we propose a new model to allow tracking transceiver-free objects. The model is based on just three communicating nodes for each area deployed on the ceiling of the tracking field as a regular triangle. This model utilizes both irregular and regular RSS dynamic information to locate the object. This approach is totally different from our previous model which only considers the ideal case. Based on our new model, the tracking accuracy can reach $1m$ by just using three nodes in a fixed pattern, while the previous methodology should use 16 nodes to reach a similar accuracy. Furthermore, using three nodes in each channel can further decrease the beacon interval of each node, resulting in a much lower tracking latency.

Our basic approach is to divide the tracking field into different triangle areas. Each triangle area is set up by three communicating nodes deployed on the ceiling of the tracking field. The adjacent triangle areas will be assigned with different channels. At first, for each triangle area, we set up a regression model. This model uses *Support Vector Regression* (SVR) to simulate the relationship between the RSS dynamics caused by the transceiver-free object and the object location on the ground. We then use this model to locate the object for each triangle area. Our experiments use TelosB [11] as communicating nodes. The experimental results show that the tracking error is around $1m$. More excitingly, the latency of our RASS can be reduced to about $0.26s$ with greater scalability and without sacrificing the accuracy.

The rest of this paper is organized as follows. In the next section, we will discuss relevant related work. Section 3 introduces the SVR tracking model and our channel assignment method, followed by the experimental results and evaluation of the performance. Finally, Section 5 concludes the paper and lists our future work.

II. Related Work

Nowadays, tracking transceiver-free objects can be divided into two categories. One is non-RF-based technologies and the other is RF-based technologies.

In non-RF technologies, video-based tracking (e.g., [7]) utilizes video streams to identify the moving objects. While this technology presents good accuracy and small latency, it has major concerns on privacy and energy consumption. Furthermore, it is costly and does not work well when the lighting condition is unsatisfactory. Smart-floor projects [6] can detect people's footsteps by using acceleration and air pressure sensors. However, these technologies require dense and careful deployment. Ultrasound [9] only has the ability to count on the number of objects passing by. While laser-based [8] technology can achieve high accuracy in its distance measurement, its cost is prohibitive nowadays.

RF-based technologies to trace transceiver-free objects are still in its infancy. Device-free passive localization [2] only pointed out the challenges lying ahead and observes radio dynamics with just 2 pairs of WiFi transceivers. While WiFi technology has the advantage in leveraging the existing infrastructure, our paper utilizes sensors, which have much lower cost and can be used to design a more dedicate tracking system. RTI [17] and motion tracking technology [10] proposed comprehensive models relating RSS measurement for a passive object. Our work focus not only the model part, but also how to utilize the model efficiently by using a proper channel assignment policy in the tracking system. Our previous work [3] [4] systematically studied basic characteristics of signal dynamics for individual wireless links when a single moving object is present. Some heuristic algorithms were proposed based on the signal changing behaviors among a group of nodes. But the model of these work only deals with the ideal case. Our new SVR model is dramatically different. It utilizes both regular and irregular RSSI dynamic information to locate the object. More importantly, our work focus on how to utilize the model efficiently by using a proper channel assignment policy, making the system scalable without sacrificing the latency and accuracy.

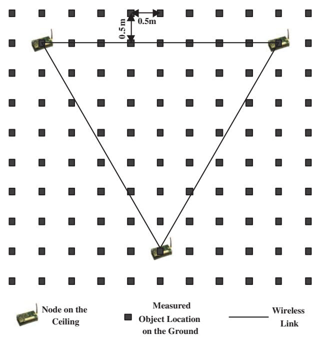


Fig. 1. Tested object locations for a 4m triangle

III. Methodology

Our RASS tracking system utilizes TelosB [11] sensor nodes in multi-channels to monitor different area of the tracking field. It can reduce the interference among the nodes belonging to different channels.

So the questions are what is the best node deployment for one channel? How many nodes should be in one channel area? What size should one channel area cover? Actually how to choose the number of nodes is a tradeoff between latency and accuracy. If we introduce many nodes in one channel area, more information may be obtained but more interference will occur among nodes to reduce the tracking accuracy. Moreover, to avoid beacon collisions, the beacon interval of each node has to be prolonged, increasing the detection latency.

Considering the above problems, we start thinking a 3-node deployment on the ceiling of the floor since 3 nodes are the minimum number of nodes to cover an area. We set them as a regular triangle because a regular shape is the best topology for the coverage problem [14]. Each

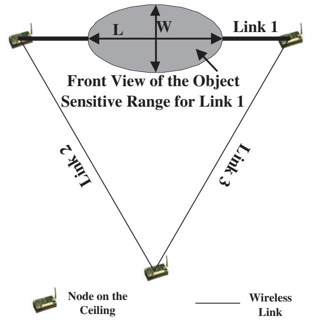


Fig. 2. Object area causing RSS dynamic for Link 1

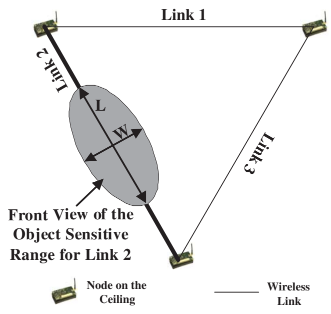


Fig. 3. Object area causing RSS dynamic for Link 2

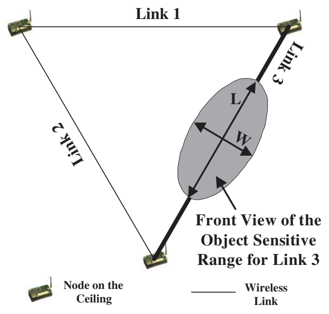


Fig. 4. Object area causing RSS dynamic for Link 3

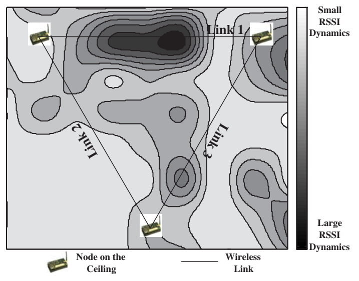


Fig. 5. RSS dynamic map for Link 1

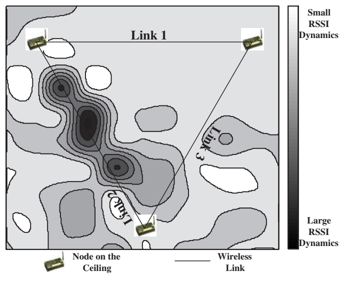


Fig. 6. RSS dynamic map for Link 2

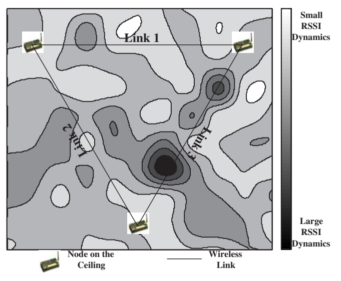


Fig. 7. RSS dynamic map for Link 3

node will periodically broadcast beacons, while receive messages from the other two nodes. Totally there are 3 pairs of links (we regard 2 nodes communicate with each other as one link). Based on the node setting, we start analysis from theoretical background. We then design a SVR model based on such deployments, which utilize both irregular and regular RSS dynamic information caused by the object to be located. At last, a novel channel assignment method is proposed.

A. Theoretical Background and Practical Cases

Under the triangle node setting on the ceiling, according to the theoretical model of two communicating nodes [3] [4] and for each link on the ceiling, there exists a defined area (proved to be an ellipse) on the ground, in which the target only exists in such an area causing a RSS dynamic. It can be calculated by $|r_d - r_s|$, where r_d is the RSS value (in dBm) when the target appears and r_s is the RSS value without the target. For example, for Link 1, when the object is in the defined area (the eclipse area colored in grey in Figure 2), it will cause a RSS dynamic of Link 1. Similar phenomena holds for Link 2 and Link 3, as shown in Figure 3 and Figure 4, respectively. If any RSS dynamic is detected, the theoretical model will assume that the object does not appear in other areas beside the 3 grey eclipses.

However, in real environments, it is not always the case. Figure 1 shows the object positions in a $4m$ triangle. Here we utilize TelosB [11] sensors and rrange a person to act as the target standing at different positions. Figures 5, 6, and 7 show how the object positions influence the RSS dynamics of Links 1, 2, 3, respectively. The place with dark color means that the object staying there will cause a large RSS dynamics and vice versa. From these figures, we may see that, in general, if an object is closer to one link, it will bring larger RSS dynamics for the corresponding link and vice versa. However, for some object positions not in the defined area, the object still may cause RSS dynamics for some links.

Thus, can we have a new and practical model just based on the regular triangle setting of only 3 nodes? Can we find a new model which can utilize the irregular information outside the model area to well locate the object position? Fortunately, SVR is such a method which is able to solve this problem and their tracking accuracy can reach around $1m$ for the $2m$, $3m$ and $4m$ triangle settings in our later experiments.

B. Model Construction using SVR

Support Vector Regression (SVR) [12] is commonly used in forecasting the financial market and reconstruction of chaotic systems. It aims to find a hyperplane which can accurately predict the training data. Considering our localization problem, we first should have some samples before tracking. In each sample, we collect both the object location and the corresponding RSS dynamics. Based on all these samples, we train a tracking model to simulate the relationship between the RSS dynamics and the object locations. We then can use this SVR model to perform prediction in the tracking state.

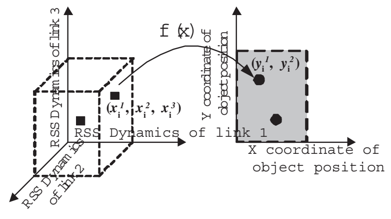
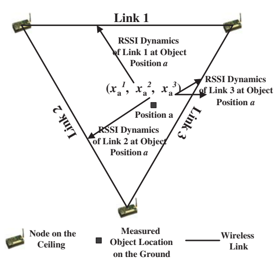
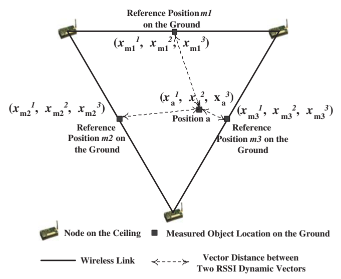


Fig. 8. SVR for localization


Link 1
RSSI Dynamics of Link 1 at Object Position a
(x_a^1, x_a^2, x_a^3)
RSSI Dynamics of Link 3 at Object Position a
Position a
Link 2
Link 3
RSSI Dynamics of Link 2 at Object Position a
Node on the Ceiling
Measured Object Location on the Ground
Wireless Link
Fig. 9. RSS dynamic vector


Reference Position m1 on the Ground
$(x_{m1}^1, x_{m1}^2, x_{m1}^3)$
$(x_{m2}^1, x_{m2}^2, x_{m2}^3)$
(x_a^1, x_a^2, x_a^3)
Position a
$(x_{m3}^1, x_{m3}^2, x_{m3}^3)$
Reference Position m2 on the Ground
Reference Position m3 on the Ground
Node on the Ceiling
Measured Object Location on the Ground
Wireless Link
Vector Distance between Two RSSI Dynamic Vectors
Fig. 10. Adaptive learning on RSS dynamics

Again, our setting is based on a regular triangle setting with 3 nodes. There are 3 links and each object position on the ground will cause RSS dynamics of each link. Suppose we have n samples: n object locations and the corresponding RSS dynamics in the triangle area. The space of the input pattern X is a 3-dimension data recording the RSS dynamics of Link 1, Link 2 and Link 3. These data are denoted by $X \in R^d, X = \{x_i^d\}, x_i^d = [x_1^d, x_2^d, ..., x_n^d]$, where d is the number of links and n is number of test object locations. In our triangle setting, $d = 3$.

The target class Y represents the object location on the ground. It is denoted by $Y \in R^k, Y = \{y_i^k\}, y_i^k = [y_1^k, y_2^k, ..., y_n^k]$, where k is dimension of object location on the ground and n is number of test object locations. In our triangle setting, $k = 2$. Given the training data $\{(x_1^d, y_1^k), ..., (x_n^d, y_n^k)\}$, our goal is to find a function $f(x) = w \cdot \Phi(x) + b, \Phi : R^d \to f, w \in R^d, b \in R$ that has at most a tolerance parameter ε from the actually obtained targets y_i^k for all the samples and at the same time is as flat as possible [12]. This means that it does not care about errors as long as they are less than the tolerance parameter ε, but it will not accept any deviation larger than this. $f(x)$ outputs the locations as shown in Figure 8.

The above method is included in the standard library LIBSVM [13]. It is utilized to train an SVR prediction model from X to Y under the triangle setting. After we get this model, when a new RSS dynamic vector is received at the tracking status, we may predict the object location by using this method.

C. Adaptive Learning with Very Small Samples

In fact, different triangles with different node distances have different RSS dynamic maps. If we choose the same triangle setting, we can set up an SVR model from one of the triangles as introduced before. This SVR model can be regarded as the general model and applied in predicting the object positions.

But if a higher tracking accuracy is preferred, we had better train different models for different triangles at different places. This, however, will be a big effort if we calibrate all the samples on each new triangle at different places. Fortunately, since same triangles have similar properties, their relationship between the RSS dynamic and the object position is also similar. We regard the RSS dynamic vector distance for each pair of object points be similar. In order to reduce such calibration efforts, we offer an adaptive learning method. In this method, only 3 samples are required for each new triangle. The others can be generated based on the old samples. Then we may use the new generated samples to refine a new SVR model for the new triangle.

1) Vector distance to reference points:

Let's look into the old model samples in triangle A. For each object location on the ground under the triangle (e.g., point a in Figure 9), its dynamic vector is $x_a = [x_a^1, x_a^2, x_a^3]$. x_a^1, x_a^2, x_a^3 are the RSS dynamics for Links 1, 2, and 3, respectively. Then, we introduce 3 reference locations $m1$, $m2$ and $m3$, as depicted in Figure 10. Their RSS dynamic vectors are $x_{m1} = [x_{m1}^1, x_{m1}^2, x_{m1}^3]$, $x_{m2} = [x_{m2}^1, x_{m2}^2, x_{m2}^3]$ and $x_{m2} = [x_{m2}^1, x_{m2}^2, x_{m2}^3]$, respectively. We choose these points as reference points because these object positions are generally cause large RSS dynamics for some links. For each position (e.g., point a), we calculate the RSS dynamic vector distance to each reference point as follows. $D_{a-i} = \sqrt{(x_a^1 - x_i^1)^2 + (x_a^2 - x_i^2)^2 + (x_a^3 - x_i^3)^2}$, here $i = m1, m2, m3$. We repeat this procedure until the distance vector between each position and each reference point is calculated.

2) Interpolation by triangulation:

Given a new triangle A' which is not trained, the shape and size of the triangle A' are the same as A. First, we test the object positions $m1'$, $m2'$, $m3'$ (the same place with $m1$, $m2$, $m3$) in triangle A' as reference points. We collect their causing RSS dynamic vectors $x_{m1} = [x_{m1}^1, x_{m1}^2, x_{m1}^3]$, $x_{m2} = [x_{m2}^1, x_{m2}^2, x_{m2}^3]$ and $x_{m3} = [x_{m3}^1, x_{m3}^2, x_{m3}^3]$. For each other object location (e.g., position a'), we may obtain its RSS dynamic vector $x_{a'} = [x_{a'}^1, x_{a'}^2, x_{a'}^3]$ by triangulating with D_{a-m1}, D_{a-m2} and D_{a-m3} obtained before.

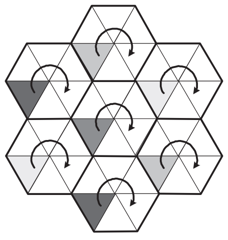


Fig. 11. Topology of Channel Assignment

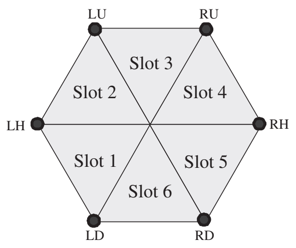
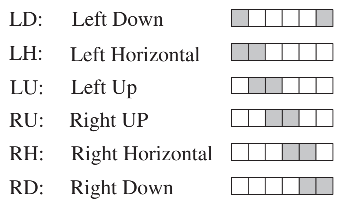


Fig. 12. Selection of the triangles in a cell



LD:	Left Down	▨	□	□	□	□	▨
LH:	Left Horizontal	▨	▨	□	□	□	□
LU:	Left Up	□	▨	▨	□	□	□
RU:	Right UP	□	□	▨	▨	□	□
RH:	Right Horizontal	□	□	□	▨	▨	□
RD:	Right Down	□	□	□	□	▨	▨

Fig. 13. Orientation-slot occupation mapping table

Repeating such a procedure, at last we may generate the RSS dynamics vector for each object location for the new triangle A'. After finishing all the interpolation, a new prediction model is constructed by using SVR introduced before. Therefore, we may easily refine a new SVR model, as long as we only measure 3 samples for the 3 reference points of the new triangle.

D. Multi-channel Assignment

Because a triangle setting is the basic tracking component, we use triangles to fully cover the whole monitored space. After deployment, we want to obtain a topology like Figure 11, where only the communication links between each two adjacent nodes exist. In order to ensure the communication topology and to avoid link interference and transmission collisions, we utilize the multi-channel ability of nodes by using a synchronized slot-based channel assignment scheme.

In our scheme, we first define a cell, which is a hexagon composed of 6 adjacent triangles as shown in Figure 12 with 7 cells in total. We define the center node in a cell as the leader and the other 6 surrounding nodes as assistants. It is obvious that one node can be either a leader or an assistant. A leader always belongs to one cell, but an assistant may belong to up to 4 adjacent cells.

We assign each cell a specific channel so that all the nodes in the same cell will use the assigned channel to communicate. The nodes are synchronized and time slots are assigned by the following scheme. For each slot at each cell, there are only 3 adjacent nodes allowed to transmit. We call the triangle formed by the 3 adjacent nodes in the same channel as a selected triangle. In our scheme, one selected triangle continues for one time slot and then changes clockwise. Thus, after six slots, all the triangles in the cell have been selected once. It is like a triangle to sweep clockwise around the cell. For example, as depicted in Figure 13, channel 1 is assigned to cell 1. The leader in the cell always stays in the same channel, while assistants only stay in channel 1 for some slots and then change to other channels to serve for other cells. Since the selection of the triangles is fixed, once an assistant knows its relative orientation to the leader, its corresponding slots for the channel are determined.

In a real deployment, the channel assignment procedure consists of three phases.

1. *Initialization.* We set up the frame of axes before deploying the nodes. After determining the location of each node, we select the cells and assign each cell a unique channel. Each node will remember its location information. If it is a leader node, it records its leader identity, the number of its assistants, and its corresponding channel for the cell. This part is done off-line.

2. *Identity determination.* Each leader broadcasts its location and its identity status to its one-hop neighbors. Once the assistant nodes have received the information from one leader, they calculate their relative orientation to the leader node and assign the leader's channel with slots according to the orientation-slot occupation mapping table. The assistant nodes then acknowledge to the leader. After confirming that the unique acknowledge number equals to the number of assistants, the leader sends a ”done” message to the sink.

3. *Synchronization.* When the sink gets all the ”done” messages, it sends out a synchronization command. Then all the nodes begin to synchronize by using the reference-broadcast method [15]. After the synchronization, the nodes enter the slot-based stage and will switch channels according to the results from step 2.

E. Number of Channels Used

To prevent interference between different hexagon cells from happening, the neighboring cells use different channels. Inside each cell, the same channel is used. Since the six inside triangles will use the channel at different time slots, there is no interference between them. According to the well-known four color map theorem [16], at least four channels should be used in our scenario. In such case, supposing in a $4m$ triangle, at one time slot, the distance between two triangles with the same channel will be at least over $8m$ apart. Consequently, the interference will be dramatically decreased. Furthermore, if we would like to avoid interference as much as possible, we may use more channels. In our experiments, TelosB nodes offer up to 16 channels, which is sufficient for our applications.

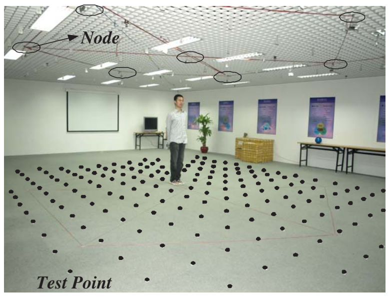


Fig. 14. Test samples

IV. Performance Evaluation

This section first shows our experimental setup and phases of the tracking system. Second, the investigation of different triangles settings is given. Third, we describe the performance of RASS system by analyzing its latency, accuracy, deployment cost. At last the tests of one moving object and multiple objects are provided.

A. Experiment Setup

Our experiment is conducted in an empty room with 20×20 square meters as shown in Figure 14. We use 10 TelosB sensors [11] with Chipcon $CC2420$ radio chips to set up 2 adjacent cells on the ceiling of the floor. Each cell is a hexagon containing 6 regular triangles. The node distance of each triangle is set as $4m$ unless otherwise specified. The default transmission power is set as $0dBm$. The radio frequency band we choose from $2400MHz$ to $2483.5MHz$. We program each node to broadcast beacons at an assigned channel in a fixed time slot as described in the last section.

The tracking algorithm has two phases. Before tracking, each node builds a static table to store the RSS values for all its neighbors in the same channel. The initialization phase is carried out in the static environment, where no target moves around. After all the nodes have built up such tables and the entire triangle areas have been swept at least once, the system enters the tracking phase. Each node measures the RSS dynamic value from different neighbors in the same channel. If the RSS dynamic value on a link is higher than some threshold (defined as the RSS dynamics in the static environment [3] as there is still some very small RSS difference in such an environment), the RSS dynamic value is reported back to the sink node. Otherwise, the dynamic value is used to update the static table.

B. Impact of the Triangle Size

Since the regular triangle is the basic tracking component in our node deployment, how to choose a suitable node distance is very important. Moreover, the tracking accuracy depends on the size of the triangle. We test from two different aspects. First, we test one pair of communicating nodes. We find when the node distance is between $2m$ and $4m$, the RSS dynamics grow larger as the object is closer to the center of the link. We call these as valid distance. For other node distances, such as smaller than $1m$ or greater than $5m$, this trend is not obvious. If the node distance is very short, the received signal strength is very strong on the line-of-sight radio propagation path and it is not easily influenced by the scatted wave caused by the object. On the contrary, if the node distance is very large, the received signal strength is very weak and is susceptible to noise interference. Second, based on the valid link range measured above, we further test on $2m$, $3m$ and $4m$ triangles with 3 communicating nodes. We find that as the node distance grows larger, there is only a little difference from their tracking errors by using our RASS algorithm. The tracking errors of different settings are all around $1m$. The experimental results for different triangles are omitted due to space limitation. If the users are not very strict with accuracy and want to save the cost, they may choose the $4m$ triangle. Hence, fewer nodes will be deployed. Otherwise, they may choose the $2m$ triangle setting.

C. Latency

The latency of RASS tracking system depends on how much time for a triangle to sweep its belonging cell since the tracking policy for each cell is independent according to our channel assignment scheme.

Each triangle contains 3 links. For one transmission direction of each link (one link contains 2 opposite transmission directions for the 2 nodes), we need to collect one RSS dynamic value once, which requires one packet transmission time. From our study [4], a TelosB sensor node takes $7ms$ on average to transmit a packet with $51bytes$. Also, our previous study revealed that the channel switching costs $0.34ms$ each time. Thus, one slot should be at least $(7 + 0.34) \times 3 \times 2 \approx 44ms$ long. Also, one hexagon cell area contains 6 triangles. It requires 6 time slot to sweep all the area. In total, it needs $44 \times 6 = 264ms \approx 0.26s$. In summary, our system can reach the real time tracking latency to as fast as about $0.26s$, which significantly outperforms previous tracking systems [3] [4] (usually 2-3s).

D. Comparative Study on Accuracy

We test different triangle sizes with $2m$, $3m$ and $4m$ node distances. Based on 42, 72 and 110 tested object positions for each triangle, we compare the tracking errors between the RASS algorithm and our previous algorithm (best-cover algorithm) [3] as shown in Figure 15. The best-cover algorithm uses 16 nodes in a $2m$ grid setting, while our current algorithm only use 3 nodes in a triangle setting. Note that we also test the best-cover algorithm in the triangle setting and the accuracy

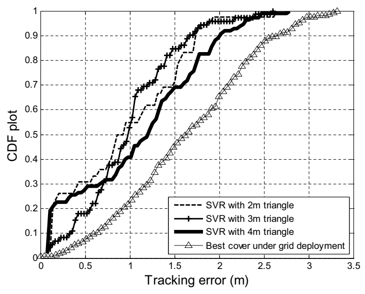


Fig. 15. Tracking error based on different triangle sizes and comparison with the best-cover algorithm

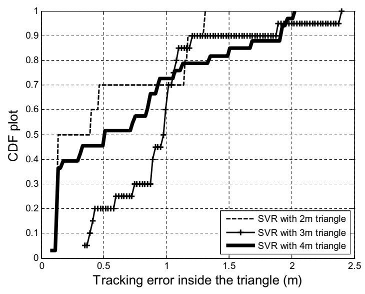


Fig. 16. Tracking error when an object is inside the triangle

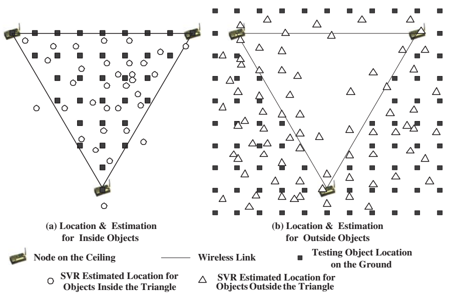

(a) Location & Estimation for Inside Objects
(b) Location & Estimation for Outside Objects
Node on the Ceiling
Wireless Link
Testing Object Location on the Ground
SVR Estimated Location for Objects Inside the Triangle
SVR Estimated Location for Objects Outside the Triangle

Fig. 17. Object locations & estimation by RASS

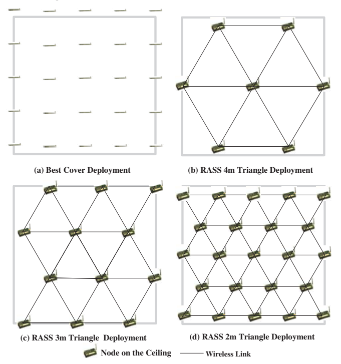

(a) Best Cover Deployment
(b) RASS 4m Triangle Deployment
(c) RASS 3m Triangle Deployment
(d) RASS 2m Triangle Deployment
Node on the Ceiling
Wireless Link

Fig. 18. Cost of deployment

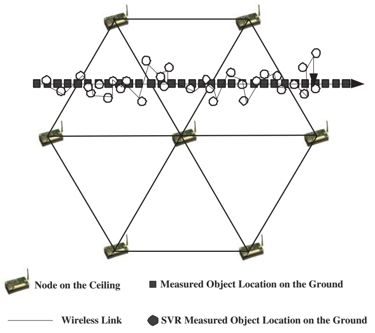

Node on the Ceiling
Measured Object Location on the Ground
Wireless Link
SVR Measured Object Location on the Ground

Fig. 19. Moving object tracking

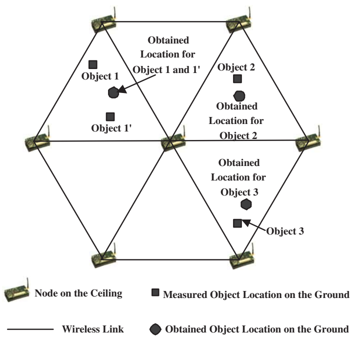

Obtained Location for Object 1 and 1'
Object 1
Object 1'
Object 2
Obtained Location for Object 2
Obtained Location for Object 3
Object 3
Node on the Ceiling
Measured Object Location on the Ground
Wireless Link
Obtained Object Location on the Ground

Fig. 20. Multiple objects

is much lower. This part is omitted here due to the space limitation. We may see that the average accuracy of our RASS algorithm is $0.98m$, $1.01m$ and $1.13m$, respectively. It greatly outperforms the best-cover algorithm, whose accuracy will decrease dramatically if using less nodes. Moreover, as the size of the triangle grows larger within the node distance limitation from $2m$ to $4m$, the tracking error of RASS system is still around $1m$.

The experiment results include the object positions inside and outside one triangle. To further investigate their respective influences on the tracking accuracy, we separate them for discussion. For the object positions which are inside one triangle, we find that 91 percent of the calculated locations by RASS are inside the triangle, as shown in Figure 17(a). For the other object positions which are outside the triangle, we find that 82 percent of the calculated locations by RASS are also outside the triangle, as shown in Figure 17(b). It means that for most object locations, we can decide it in the right triangle.

We can draw the conclusion that for a single object to be tracked, it has a high probability to be recognized inside the right triangle. As Figure 16 depicts, no matter based on $2m$, $3m$ or $4m$ triangles, 70 percent of the tracking errors is under $1m$ for the object inside the triangle. So if two or more adjacent triangles all estimate one object inside them at the same time, we average their coordinates as the output object location. Such case often happens when the object is around the border area connecting many adjacent triangles. If only one triangle senses the object inside it, we just output it. Moreover, since a moving object usually has a trace passing many triangles, we may easily calibrate its trace to eliminate some abnormal estimated locations.

E. Cost of Deployment

The RASS tracking system can give users flexible choices with deployment. If they do not care the slight difference of tracking accuracy, they may choose the $4m$ triangle setting to save deployment cost. Its tracking accuracy still can reach around $1m$. Otherwise, they may choose the $2m$ triangle setting. For example, as shown in Figure 18, suppose we are in an 8×8 square meter tracking field. The best-cover algorithm [3] requires 25 nodes deployed in this area, where the RASS system only needs at least 7 ($4m$ triangle) and at most 23 ($2m$ triangle) to reach a similar accuracy with significantly improved latency. The proposed RASS tracking system can save from 8% up to 72% deployment cost in total.

F. Tracking Moving Objects

The RASS tracking system has a good ability to track moving objects because the time for one triangle area to finish sweeping one cell is $0.26s$, It is the same as the tracking latency for the whole field. The time of the server to predict the object position is negligible. So every $0.26s$, we have a report for the object location. To study the effect of one moving object, we arrange a person to walk through a fixed trace under the cells.

We choose the $4m$ triangle setting inside each cell. The person's moving speed is around $1m/s$. One of the testing trace is shown in Figure 19. Its average tracking error is $0.87m$. If the object always moves along the border area, the tracking accuracy will decrease, which is left for the future work.

G. Multiple Objects

The RASS system can trace multiple objects, if they are in different triangles and faraway from each other. As the size of the rectangle can be chosen by different users, deploying small size triangles can help to separate different objects. Our experiment shows that the smallest node distance of the triangle is $2m$. Based on such setting, we can reach the tracking accuracy of each object to $0.98m$ in average. Therefore, if the deployment cost is not the primary concern and we care more about the tracking accuracy, the $2m$ triangle deployment is recommended. The area of the $2m$ triangle is about 1.73 square meters. Hence, only if the locations of the multiple objects are beyond this limit, we may locate them (e.g., object 2 and object 3 in Figure 20). Otherwise, we just regard them as one big object. As Figure 20 shows, object 1 and object $1'$ are in the same triangle. We regard them as one object.

V. Conclusion and Future Work

This paper makes a completely new design of transceiver-free object tracking RASS. It is dramatically different from our previous work. First, we proposed an SVR model (based on a triangle setting with 3 nodes) to use both irregular and regular RSS dynamic information caused by the target, while our previous model only considers the ideal case. Our experimental results showed that the SVR model can exhibit the same tracking accuracy with dramatically reduced number of nodes comparing with the previous work. Using 3 nodes can achieve the same accuracy as requiring 16 nodes in previous approaches. Second, we proposed a multiple-channel assignment algorithm to well utilize the SVR model. Our basic idea is to divide the tracking field into different triangle areas. Each triangle will use different channels at different time slots. For each triangle area, we utilize the SVR model to estimate the object position. By using this scheme, we can avoid the interference between different channel areas and the latency can be dramatically reduced. As far as we know, we are the first to introduce multiple-channel assignment in the tracking area. Transceiver-free object tracking is one of the scenarios. It can be easily extended to the other applications. With the new design, the accuracy, latency and cost of the RASS system can all be significantly improved. It can achieve a good tracking accuracy around $1m$ with the latency bounded by $0.26s$. The deployment cost can be reduced by 72%. Our system is very scalable to a large deployment without sacrificing the latency and accuracy.

As future work, we want to try a larger area with different node topologies. Our solution to multiple moving objects is still limited by the size of the rectangles. If the objects are all in the border area, the tracking error will increase. A better model has to be found for the objects which are very close together.

Acknowledgement

This research was supported in part by Hong Kong RGC Grant HKUST617710, China NSFC Grants 60933011, 60933012 and 60673122, the Science and Technology Planning Project of Guangdong Province, China under Grant 2009A080207002, the SZ-HK Innovation Circle Project under Grant ZYB200907060012A, China NSFGD Grant 10351806001000000 and the Science and Technology Project of Shenzhen JC200903120046A.

References

[1] S. N. Patel, J. A. Kientz, G. R. Hayes, S. Bhat and G. D. Abowd, "Farther Than You May Think: An Empirical Investigation of the Proximity of Users to Their Mobile Phones", in *Proceeding of ACM UbiComp*, 2006.

[2] M. Youssef, M. Mah and A. Agrawala, "Challenges: device-free passive localization for wireless environments", in *Proceeding of ACM MobiCom*, 2007.

[3] D. Zhang, J. Ma, Q. Chen, and L. M. Ni , "An RF-based system for tracking transceiver-free objects", in *Proceeding of IEEE PerCom*, 2007.

[4] D. Zhang, L. M. Ni,"Dynamic Clustering for tracking multiple transceiver-free objects", in *Proceeding of IEEE PerCom*, 2009.

[5] [Online]. Available: http://www.acorel.com.

[6] J. O. Robert and D. A. Gregory, "The smart floor: a mechanism for natural user identification and tracking", in *Proceeding of ACM CHI*, 2000.

[7] Q. Cai and J. K Aggarwa, "Automatic tracking of human motion in indoor scenes across multiple synchronized video streams", in *Proceeding of IEEE ICCV*, 1998.

[8] R. Dorsch, G. Hausler, J. Herrmann, "Laser triangulation: fundamental uncertainty in distance measurement", in *Applied OPTICS*, 1994.

[9] Q. Chen, M. Gao, J. Ma, D. Zhang, L. M. Ni, Y. Liu, "moving object counting using ultrasonic sensor networks", in *International Journal of Sensor Networks*, 2008.

[10] J. Wilson, N. Patwari, "See Through Walls: Motion Tracking Using Variance-Based Radio Tomography Networks", in *Proceeding of IEEE TMC*, 2010.

[11] XBOW Corporation, "TelosB mote specifications" http://www.xbow.com/Products/productdetails.aspx?sid=252.

[12] A. J. Smola and B. Scholkopf, "A tutorial on support vector regression", in *Statistics and Computing 14(3):199-222*, 2004.

[13] [Online]. Available: http://www.csie.ntu.edu.tw/ cjlin/libsvm/

[14] S. M. Nazrul Alam and Zygmunt J. Haas, "Coverage and Connectivity in Three-Dimensional Networks", in *Proceeding of ACM MobiCom*, 2006.

[15] J. Elson, L. Girod and D. Estrin, "Fine-Grained Network Time Synchronization using Reference Broadcasts", in *Proceeding of OSDI*, 2002.

[16] [Online]. Available: http://mathworld.wolfram.com/Four-ColorTheorem.html.

[17] J. Wilon and N. Patwari, "Radio Tomographic Imaging with Wireless Networks", in *Proceeding of IEEE TMC*, 2010.