6-5-2014

On Non-Cooperative Multiple-Target Tracking with Wireless Sensor Networks

Ye Zhu
Cleveland State University, y.zhu61@csuohio.edu

A. Vikram
Cleveland State University

Huirong Fu
Oakland University, fu@oakland.edu

Yong Guan
Iowa State University

On Non-Cooperative Multiple-Target Tracking With Wireless Sensor Networks

Ye Zhu, *Member, IEEE*, Anil Vikram, Huirong Fu, *Member, IEEE*, and Yong Guan, *Senior Member, IEEE*

***Abstract*—In this paper, we propose an approach to track multiple non-cooperative targets with wireless sensor networks. Most existing tracking algorithms can not be directly applied to non-cooperative target tracking because they assume the access to signals from *individual* targets for tracking by assuming that: 1) there is only one target in a field; 2) signals from different cooperative targets can be differentiated; or 3) interference caused by signals from other targets is negligible because of attenuation. We propose a general approach for tracking non-cooperative targets. The tracking algorithm first separates the *aggregate* signals from *multiple indistinguishable* targets via the blind source separation (BSS) algorithms. Through the analysis on both the temporal and spatial correlation of the *separated individual* signals, the tracking algorithm determines the location of a target and its moving track. A voting scheme based on the spatial information is designed to better estimate the moving track. Furthermore, we analyze and discuss the influence of signal attenuation and the tracking resolution of the proposed tracking approach. Our experiments show that the proposed approach can both accurately and precisely track multiple indistinguishable moving targets.**

***Index Terms*—Tracking, blind source separation (BSS), wireless sensor networks.**

I. Introduction

TRACKING moving targets is one of the prominent applications of wireless sensor networks. Sensors, also called as "smart dust," are small devices known for their simplicity and low cost. Using a network of sensors with wireless communication capabilities enables both cost-effective and performance-effective approaches to track targets, due to the availability of a large amount of data collected by sensors for tracking targets. Depending on the applications, sensors with different sensing modalities such as acoustic, seismic, infrared, radio, and magnetic can be deployed for tracking different types of targets.

In this paper, we are particularly interested in applying wireless sensor networks in non-cooperative tracking, which means that targets are not intentionally participating in the tracking. The examples are tracking targets in hostile environments such as enemy soldiers carrying radios in a battlefield and tracking wild animals with acoustic sensors. The first challenge in the non-cooperative tracking comes from the fact that data collected by sensors is *aggregate* data. In other words, signals received by sensors are generally *mixtures* of signals from *individual* targets. For example, a sensor in a field of interest may receive signals from more than one target. Obviously tracking targets based on the mixture signals can result in inaccurate results when interference from targets other than the one of interest is not negligible. For brevity, we use the term *aggregate signal* to mean the signal received by sensor, i.e., data collected by sensors and *individual signal* to mean the signal transmitted from or caused by individual targets in the rest of the paper. For cooperative targets, the challenge can be easily overcome since sensors can distinguish the cooperative targets by tags embedded in signals or by having different targets to send signals using different channels such as using different frequency bands. For non-cooperative tracking, signals from non-cooperative targets such as wild animals or radios carried enemy soldiers do not carry any identifiers originally or have identifiers removed on purpose. So in non-cooperative target tracking, targets are indistinguishable and it is impossible to have the direct access to the *individual signals*.

Because of the challenge, most existing tracking algorithms for wireless sensor networks are not suitable for the non-cooperative tracking due to the following limitations:

- Single-target: Many tracking algorithms assume that only one target exists in a field of interest. So signals received by sensors are essentially *individual signals*.
- Negligible interference: Some researches assume that interference from targets other than the one of interest is negligible. The assumption is legitimate only for applications in which signal from a target attenuates dramatically when the distance between the target and sensor increases.
- Distinguishable targets: Some researches assume that sensors can distinguish targets by tags embedded in signals or by having different targets to send signals using different channels such as using different frequency bands.

We propose an approach based on Blind Source Separation, a methodology from statistical signal processing to recover unobserved "source" signals from a set of observed mixtures of the signals. Blind source separation models were originally defined to solve the *cocktail party problem*: The blind source separation algorithms can extract one person's voice signal from given mixtures of voices in a cocktail party. Blind source

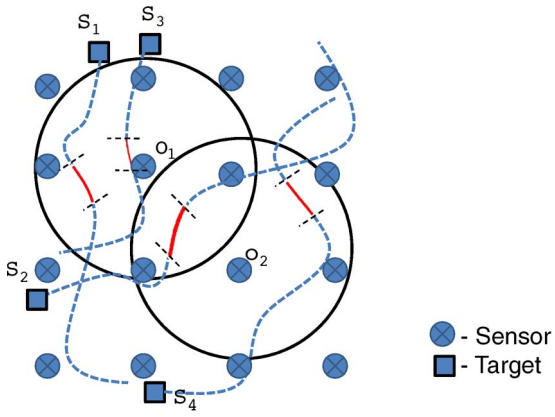

Fig. 1. Network model. In the figure, the dashed lines represent the targets' moving paths and the red solid parts of the moving paths represent path fragments that targets are moving on simultaneously. The solid circles represent sensing ranges of Sensors O_1 and O_2.

separation algorithms solve the problem based on the independence between voices from different persons. Similarly, in the target-tracking problem, it is generally safe to assume *individual signals* from different targets are independent. So, we can use blind source separation algorithms to recover *individual signals* from *aggregate signals* collected by sensors. For the cases in which *individual signals* are dependent, blind source separation algorithms based on timing structure [1] of *individual signals* can be used. To our knowledge, we are the first to apply the Blind Source Separation (BSS) algorithms to tracking indistinguishable moving targets in wireless sensor networks. However, simply applying the BSS algorithms can not solve the tracking problem since the output of BSS algorithms includes not only recovered individual signals, but also noise signals, aggregate signals, and partial signals, which contain part of individual signals in different time durations. Therefore, we propose a clustering and selecting algorithm to remove noise and artifacts introduced by BSS algorithms. We also design a voting algorithm based on the signal spatial information to further improve the performance.

Currently we limit our researches on tracking targets for outdoor applications such as tracking animals in open areas and tracking soldiers in open battle fields or tracking in large indoor facilities such as stadiums. The algorithm does not work well for indoor applications mainly because of the multi-path effect.

The rest of the paper is organized as follows: Section II outlines our network model and assumptions. The main idea of the non-cooperative tracking based on BSS algorithms is described in Section III. In Section IV, we describe our approach in details. We theoretically analyze the performance of our approach and effect of parameters used in our approach in Section V. The evaluation of our approach is presented in Section VI. In Section VII, we review related work. The paper concludes in Section IX.

II. Network Model and Assumptions

The goal of this project is to track multiple targets with wireless sensor networks as shown in Fig. 1. Wireless sensors are randomly deployed in the field of interest. Generally, a wireless sensor can receive a mixture of individual signals from multiple sources. For example, suppose a number of sensors are deployed in Fig. 1, Sensor O_1 can receive signals from Targets S_1, S_2, and S_3 during one time duration. Following are the assumptions made in this general model: (1) Sensors have no capability to distinguish targets. This assumption is important for deploying sensors in uncooperative or hostile environments such as tracking enemy soldiers with wireless sensor networks. (2) The location of each sensor in the sensor network is known. Location information can be gathered in a variety of ways. For example, the sensors may be planted, and their locations are marked. Alternatively, sensors may have GPS capabilities. Finally, sensors may locate themselves through one of several schemes that rely on sparsely located anchor sensor nodes [2]. (3) Aggregate signals collected by wireless sensors can be gathered for processing by a sink or gateway. Data compression or coding schemes designed for sensor networks such as ESPIHT [3] can be used to reduce the data volume that is caused by spatial redundancy across neighboring nodes or temporal redundancy at individual nodes. (4) Targets are moving under a speed limit. Obviously it is impossible to track a high-speed target that only generates a small amount of data when passing the field of interest. We analyze the effect of the speed limit in Section V.

III. Recovering Individual Signals for Target Tracking

In this section, we introduce blind source separation and the rationale of applying blind source separation to the multiple target tracking problem using wireless sensor networks.

A. Theoretical Foundation: Blind Source Separation

Blind Source Separation (BSS) is a methodology used in statistical signal processing to recover unobserved "source" signals from a set of observed mixtures of the signals. The separation is called *blind* to emphasize that the source signals are not observed and that the mixture is a black box to the observer. While no knowledge is available about the mixture, in many cases it can be safely assumed that source signals are independent. In its simplest form [4], the blind source separation model assumes n independent signals $S_1(t), \ldots, S_n(t)$ and n observations of mixture $O_1(t), \ldots, O_n(t)$ where $O_i(t) = \sum_{j=1}^{n} a_{ij} S_j(t)$. The goal of BSS is to reconstruct the source signals $S_j(t)$ using only the observed data $O_i(t)$, based on the assumption of independence among the signals $S_j(t)$. Given the observations $O_i(t)$, BSS techniques estimate the signals $S_j(t)$ by maximizing the independence between the estimated signals. The common methods employed in blind source separation are minimization of mutual information [5], maximization of nonGaussianity [6], and maximization of likelihood [7].

B. Recover Individual Signals for Target-Tracking With Blind Source Separation Algorithms

In our tracking approach, blind source separation algorithms are used to recover *individual signals* (i.e., source signals as in

the BSS literature introduced in Section III-A) from *aggregate signals* (i.e., observations as in the BSS literature introduced in Section III-A). Suppose a number of sensors are deployed in the field as shown in Fig. 1, Sensor O_1 can receive signals from Targets S_1, S_2, and S_3 and Sensor O_2 can receive signals from Targets S_2 and S_4. With a slight abuse of the notation, we represent the signal received by Sensor O_i as $O_i(t)$ and the individual signal from Target S_i as $S_i(t)$. So we can have the following two equations: $O_1(t) = a_{11}S_1(t) + a_{12}S_2(t) + a_{13}S_3(t)$, $O_2(t) = a_{22}S_2(t) + a_{24}S_4(t)$, where a_{ij} denotes the attenuation from Target S_j to Sensor O_i. In general, for m neighboring sensors and n targets, we can rewrite the problem in vector-matrix notation,

$$\begin{pmatrix} O_1(t) \\ O_2(t) \\ \vdots \\ O_m(t) \end{pmatrix} = \mathbf{A}_{m\times n} \begin{pmatrix} S_1(t) \\ S_2(t) \\ \vdots \\ S_n(t) \end{pmatrix} \quad (1)$$

where $\mathbf{A}_{m\times n}$ is the matrix of attenuation between targets and sensors which can be modeled as the *mixing matrix* in the BSS literature. Given the observations $O_i(t)$, BSS algorithms can recover the source signals $S_j(t)$ without knowledge on the mixing matrix. Since the individual signals are independent from each other—they come from different targets—we can use any of the algorithms mentioned in Section III-A to recover individual signals $S_1(t), \ldots, S_n(t)$.

While the goal of BSS in this context is to re-construct the original signals $S_i(t)$, in practice the separated signals are sometimes only loosely related to the original signals. We categorize these separated signals into four types, as follows: In the first case, the separated signal is correlated to actual individual signals $S_i(t)$. The separated signal in this case may have a different sign than the original signal. We call this type of separated signal an *individual separated signal*. In the second case, a separated signal may be correlated to an *aggregate of signals* from several targets. This happens when signals from more than two targets can be "heard" by all the sensors. In such a case, the BSS algorithm would not be able to fully separate the signal mixture into the individual separated signals. Rather, while some individual signals can be successfully separated, others remain aggregated. In the third case, separated signals may be correlated to one original signal in the beginning part and correlated to another original signal in the rest. We call this type of separated signal a *partial separated signal*. This happens when a target moves out of one sensing range and enters into another sensing range. In the fourth case, separated signals may represent *noise signals*. Noise signals are separated out when neighboring sensors receive different individual signals from the same target. The difference can be caused by signal attenuation or interference. BSS algorithms separate the difference as noise signals. The effect of signal attenuation on separation performance is described in Section V-A.

IV. Tracking Algorithm

The main idea of the tracking algorithm is as follows: First, individual signals are recovered from the aggregate signals observed by the sensors with BSS algorithms. Given the recovered individual signals, the tracking algorithm then intersects sensing ranges of sensor groups "hearing" the same individual signal to locate targets in one time duration. The tracking algorithm outputs the path taken by a target by linking the locations estimated by intersecting the sensing ranges. The tracking algorithm consists of six steps: (1) In the preparation step, aggregate signals collected from sensors are grouped in the space domain and segmented in the time domain and these groups of signal segments are fed to the second step, the blind source separation step. (2) BSS algorithms are applied to aggregated signals collected from each sensor group and output separated signals. As described in Section III, these separated signals contain individual separated signals, aggregate separated signals, noise signals, and partial separated signals. The following clustering step and selection step are designed to remove noise signals, partial separated signals, aggregate separated signals. (3) The clustering step will cluster these separated signals. (4) The selection step selects *individual separated signals*, i.e., separated signals that are closest to actual individual signals from clusters formed in the clustering step. (5) The intersection step estimates fragments of paths by geographically intersecting the sensing ranges of sensor groups that can "hear" the same *individual separated signal*. (6) The voting step outputs estimated paths by linking path fragments generated in the intersection step into paths. The details of these six steps (preparation, separation, clustering, selection, intersection, voting) are described below.

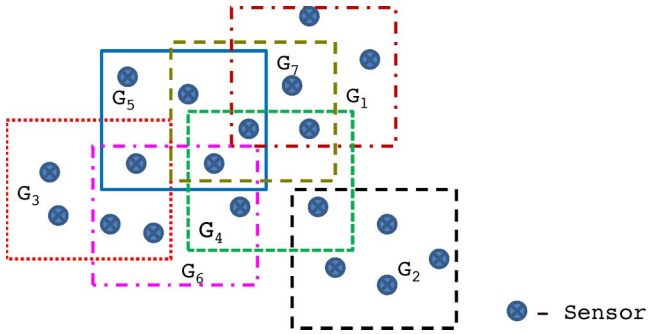

Fig. 2. Grouping ($n_{\text{group}} = 5$). In the figure, sensors within a rectangular form a sensor group.

A. Preparation Step

To fully utilize information collected from wireless sensor networks, aggregate signals collected by wireless sensors are grouped spatially and segmented temporally. As shown in Fig. 2, sensors in the field are grouped into sensor groups. For each sensor, a sensor group of n_{group} sensors is formed with its closest sensors. So a sensor may be included in multiple sensor groups and the sensing area of a sensor group vary depending on the location of the sensors in the sensor group. Aggregate signals collected from each sensor group are segmented according to time slots shown in Fig. 3. Each time slot is of length l_{seg} and so the segment in a time slot is of the *segment length* l_{seg}. The step size, defined as the difference between the starting positions of two successive time slots, is denoted as l_{step}. So two successive signal segments have a common part of length $l_{\text{seg}} - l_{\text{step}}$. A BSS algorithm will be applied on grouped aggregate signals sequentially, i.e., segment by segment in the

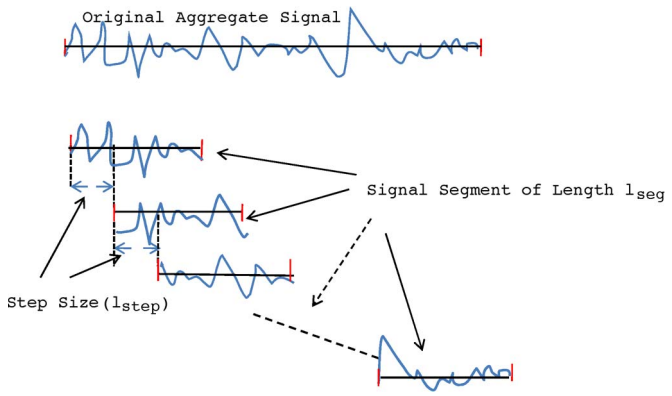

Fig. 3. Signal segments.

next step. We represent the segment group from the ith sensor group and the jth time slot as $OG_{i,j}$. The pth segment in the group is denoted as $O^p_{i,j}$. In set theory notation, $OG_{i,j} = \{O^p_{i,j} : p = 1, \ldots, n_{\text{group}}\}$. The output of the preparation step is segment groups $OG_{i,j}$.

Spatial redundancy and temporal redundancy are created during the grouping and the segmenting, respectively. We use the term, spatial redundancy, to mean the fact that a sensor can be grouped into more than one sensor groups. The temporal redundancy is created in segmenting since two successive time slots have overlaps. Both spatial redundancy and temporal redundancy are created so that noise and artifacts possibly generated in the following separation step can be eliminated in the clustering and the selection step.

After the preparation step, signals are all in unit of segments. We use actual segments, individual segments, aggregate segments, partial segments, noise segments to mean segments of original individual signals, individual separated signals, aggregate separated signals, partial separated signals, and noise signals respectively in the rest of the paper.

B. Separation Step

In the separation step, a BSS algorithm is applied on segments contained in $OG_{i,j}$ for all i and j. The separation step outputs groups of separated segments denoted by $SG_{i,j}$, i.e., the group of segments separated from $OG_{i,j}$.

C. Clustering Step

The clustering step is designed to form clusters so that the following selection step can select *individual segments* from the clusters. Essentially the clustering eliminates noise segments, aggregate segments, and partial segments from processing in the following steps. The clustering step takes advantage of spatial redundancy created in the preparation step. The rationale behind this step is as follows: if a separated signal represents an individual signal, similar signals will be separated in at least similar forms by more than one neighboring sensor groups. In contrast, a separated signal that was generated because of attenuation or interference is unlikely to be generated by more than one group.[1] In our experiments, agglomerative hierarchical clustering [8] is used. Based on the rationale, we use correlation as the measure of similarity, and define the distance between two separated segments as follows:

$$D\left(S'^p_{i,j}, S'^q_{k,j}\right) = 1 - \left|\text{corr}\left(S'^p_{i,j}, S'^q_{k,j}\right)\right| \quad (2)$$

where $S'^p_{i,j}$ denotes the pth segment in separated segment group $SG_{i,j}$, and $corr(X, Y)$ denotes the correlation coefficient of segments $X = [x_1, x_2, \ldots, x_{l_{\text{seg}}}]$ and $Y = [y_1, y_2, \ldots, y_{l_{\text{seg}}}]$. The correlation coefficient can be calculated as follows:

$$corr(X,Y) = \frac{\sum_{i=1}^{l_{\text{seg}}}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{l_{\text{seg}}}(x_i - \bar{x})^2 \sum_{i=1}^{l_{\text{seg}}}(y_i - \bar{y})^2}}$$

where $\bar{x} = (\sum_{i=1}^{l_{\text{seg}}} x_i)/l_{\text{seg}}$ and $\bar{y} = (\sum_{i=1}^{l_{\text{seg}}} y_i)/l_{\text{seg}}$. We use the absolute value of the correlation coefficient because the separated segments may be of different sign than the actual segment. Clustering will only cluster segments of the same time slots as indicated in the distance measure defined in Equation (2). The number of clusters formed in this step is heuristically set to two times the number[2] of targets in the field because some clusters may contain only partial segments and noise segments. These clusters of partial segments and noise segments are removed in the following selection step. The complexity of the clustering step is essentially the complexity of the agglomerative hierarchical clustering algorithm, which is $O(\rho^3)$, where ρ denotes the number of separated segments.

The input of the clustering step is $SG_{i,j}$ and the clustering step outputs clusters formed in each time slots. We use $Clst^i_j$ to denote the ith cluster formed in the jth time slot.

D. Selection Step

The goal of the selection step is to select *individual segments*, i.e., separated signal segments that are closest to actual individual signal segments, from clusters formed in the previous step. One naive approach is to simply select the segment at the center of each cluster according to the distance defined in Equation (2). In our experiments, we find the approach is not robust since aggregate segments and partial segments may also be clustered into the clusters of individual segments.

The selection step is based on the temporal redundancy created in the preparation step. For ease of understanding, we use the example in Fig. 4 to describe the rationale behind the selection step. Because of the overlap between the sensing range of the ith sensor group and the sensing range of the kth sensor group, both sensor groups are able to "hear" the target at the same time. Without loss of generality, we assume both sensor groups can "hear" the target from the $j+1$th time slot. Because of the temporal redundancy as described in Section IV-A, the $j+1$th time slot has a common part of length $l_{\text{seg}} - l_{\text{step}}$ with the jth time slot. In turn, the signal received from the target by the ith sensor group during the jth time slot has a *common* part with the signal received from the same target by

[1] More analysis of attenuation and interference can be found in Section V.

[2] The number of targets can be either known *a priori* or can be estimated using existing algorithms [9]–[11]. Similar tracking performance was observed with more clusters mainly because of the following selection step.

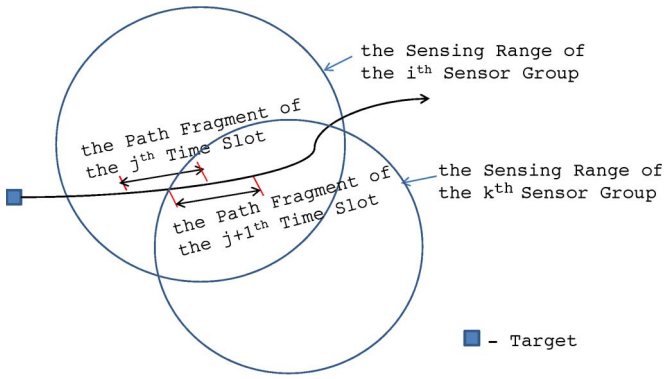

Fig. 4. Rationale behind the selection step.

the kth sensor group during the $j+1$th time slot. So one of the separated segments from the ith sensor group and jth time slot, denoted as[3] $S'^{p}_{i,j}$, should be similar as one separated segment from the kth sensor group and $j+1$th time slot, denoted as $S'^{q}_{k,j+1}$. To measure the similarity, we define the *link correlation* between the two separated segments as follows:

$$\rho\left(S'^{p}_{i,j}, S'^{q}_{k,j+1}\right) = \left|\mathrm{corr}\left(S'^{p}_{i,j}(l_{\mathrm{step}}, l_{\mathrm{seg}}), S'^{q}_{k,j+1}(0, l_{\mathrm{seg}} - l_{\mathrm{step}})\right)\right| \quad (3)$$

where $S'^{p}_{i,j}(x,y)$ denotes the part of Segment $S'^{p}_{i,j}$ from the xth data sample to the yth data sample and $\rho(S'^{p}_{i,j}, S'^{q}_{k,j+1})$ denotes the link correlation between segments $S'^{p}_{i,j}$ and $S'^{q}_{k,j+1}$. Absolute value is used in link correlation definition because the separated segments may be of different sign than the actual segments. Using the example in Fig. 4, we can see that the link correlation functions as connected links in a chain that chains the path segments of the same target.

The example in Fig. 4 shows the case when a target is moving along a path. If the target is static within the sensing range of the ith sensor group, then the link correlation $\rho(S'^{p}_{i,j}, S'^{q}_{i,j+1})$ should be high. In other words, one separated segment from the ith sensor group and the jth time slot, denoted as $S'^{p}_{i,j}$, should be very similar as one of the separated segments from the same sensor group and the $j+1$th time slot, denoted as $S'^{q}_{i,j+1}$. To generalize the two cases, we redefine the link correlation as follows:

$$\rho\left(S'^{p}_{j}, S'^{q}_{j+1}\right) = \left|\mathrm{corr}\left(S'^{p}_{j}(l_{\mathrm{step}}, l_{\mathrm{seg}}), S'^{q}_{j+1}(0, l_{\mathrm{seg}} - l_{\mathrm{step}})\right)\right| \quad (4)$$

where S'^{p}_{j} denotes the pth separated segment from the jth time slot among segments separated from all sensor groups.[4]

The selection step can prevent segments of noise-segment clusters and partial-segment clusters from being selected since noise and artifact generated by separation algorithms in one time slot will unlikely be generated again in the following time slot. To make the algorithm more robust, we extend the link correlation defined in Equation (4), which is based on two consecutive time slots. The extended link correlation, defined below, is based on n_{slot} consecutive time slots (CTS):

$$\mathrm{P}_{\mathrm{CTS}_j}\left(S'^{x_j}_{j}, S'^{x_{j+1}}_{j+1}, \cdots, S'^{x_{j+n_{\mathrm{slot}}}}_{j+n_{\mathrm{slot}}}\right) = \sum_{i=j}^{j+n_{\mathrm{slot}}-1} \rho\left(S'^{x_i}_{i}, S'^{x_{i+1}}_{i+1}\right) \quad (5)$$

where CTS_u denotes the uth CTS containing time slots $\{u, u+1, \ldots, u+n_{\mathrm{slot}}-1\}$. Essentially the link correlation defined in Equation (5) is the sum of the link correlation (defined in Equation (4)) calculated for n_{slot} consecutive time slots. In each time slot, the K individual segments with top K sum of link correlation defined in Equation (5) are selected. Only one individual segment will be selected from a cluster. The number of individual segments selected in each time slot is K, the number of targets in the field. The value of K is either known *a priori* or can be estimated by using existing algorithms [9]–[11].

The pseudo code of the selection step can be found in [12]. The input to the selection step is $Clst^{i}_{j}$ and the output is selected individual segments $C^{u}_{k,j}$ that denotes the kth individual segment selected for jth time slot based on CTS_u.

E. Intersection Step

The intersection step estimates one fragment of a path based on each individual segment selected in the previous step. One path fragment is estimated by geographically intersecting the sensing ranges of sensor groups that can "hear" the same target. Since individual segments are segments most "resembling" to the original individual segments from targets, the sensor groups which can "hear" the same target can be found as follows: For one selected individual segment $C^{u}_{k,j}$ (denoting the kth individual segment selected for the jth time slot based on CTS_u), if a sensor group has one separated segment S'^{m}_{j} (denoting the mth separated segment among all the segments separated in the jth time slot) highly correlated to the selected individual segment $C^{u}_{k,j}$, the sensor group is determined as a sensor group which can "hear" a target.

Since the estimation is based on intersecting sensing ranges of sensor groups that can "hear" the same target. The order of sensing ranges being intersected is important to the estimation accuracy. The order used in this step is determined by the absolute value of the correlation with the selected individual segment $C^{u}_{k,j}$, the output of the selection step. In other words, the sensor group that has separated segments more similar to the selected individual segment $C^{u}_{k,j}$ will have its sensing range intersected earlier.

The input of this step is selected individual segments $C^{u}_{k,j}$. For each selected individual segment, an intersection area $\mathrm{area}^{u}_{k,j}$ is generated as output of this step. These generated areas are estimated path fragments. The pseudo code for the intersection step can be found in [12].

F. Voting Step

The voting step concatenates the "best" path fragments estimated in the previous step to form an estimated path. The "best"

[3]To differentiate separated signals from original individual signals, we use S' to denote separated signals and S to denote original individual signals.

[4]We remove i the index of sensor groups, from $S'^{p}_{i,j}$, since link correlation can be calculated for different sensor groups and the same sensor group. In the rest of the paper, we use S'^{p}_{j} to denote the pth segment separated from the jth time slot among segments separated from all sensor groups.

path fragments are selected by a voting mechanism. Before explaining the details of the voting mechanism, we would like to first introduce the distance metric d_{area} which measures the distance between two estimated path fragments, i.e., two intersection areas output by the intersection step. The distance $d_{\text{area}}(\text{area}^u_{X,j}, \text{area}^u_{Yj+1})$, i.e., the distance between two path fragments denoted by $\text{area}^u_{X,j}$ and $\text{area}^u_{Y,j+1}$, is defined as the minimum distance between any two points from these two path fragments respectively. So if the two path fragments overlap with each other, then $d_{\text{area}}(\text{area}^u_{X,j}, \text{area}^u_{Y,j+1}) = 0$.

The voting mechanism takes advantage of the temporal redundancy created in the preparation step. The "best" path fragment selected to form an estimated path should satisfy the following two requirements: (1) The selected path fragment $\text{area}^u_{k,u}$ should have zero distance with all path fragments estimated based on the same CTS_u, i.e., $dcur_{k,u} = \sum_{j=u}^{u+n_{\text{slot}}-2} d_{\text{area}}(\text{area}^u_{k,u}, \text{area}^u_{k,j+1}) = 0$. (2) The selected path fragment should also have zero distance with all the path fragments estimated for the same time slot based on different CTS, i.e., $dpre_{k,u} = \sum min(d_{\text{area}}(\text{area}^u_{k,u}, \text{area}^{u-m}_{1,u}), \ldots, d_{\text{area}}(\text{area}^u_{k,u}, \text{area}^{u-m}_{K,u})) = 0$.

Finally a path is estimated by linking path fragments selected from different time slots. To determine whether a selected "best" path fragment, say $Pseg_{z_j,j}$ denoting the z_jth selected path fragment for the jth time slot, belongs to a path, say $epath_l$ denoting the lth estimated path, the distance between $Pseg_{z_j,j}$ and the path fragment of $epath_l$ determined in the previous time slot, say $Pseg_{z_{j-1},j-1}$, is calculated. If the distance is zero, then $Pseg_{z_j,j}$ is determined as one path fragment of $epath_l$. The pseudo code of the voting step is shown in Appendix A.

V. Analysis of the Tracking Algorithm

In this section, we analyze the effect of signal attenuation and the tracking resolution.

A. Signal Attenuation

Signal attenuation is a natural consequence of signal transmission over long distances. It is a function of transmission distance.

When static targets are being tracked, signal attenuation will not affect tracking performance. Since targets are static, the distance between targets and sensors does not change over time. So the attenuation can be modeled as a constant. For the same individual signal from a target, different sensors will observe different attenuation because of different transmission distance. So individual signals received by different sensors from the same target are different only by a scaling factor. The difference because of the scaling factor can be absorbed by the mixing matrix defined in the BSS model as Equation (1). So attenuation does not affect tracking static targets by our approach.

When moving targets are being tracked, signal attenuation may cause noise signals in the output of the separation step. When targets are moving, the difference between individual signals received by different sensors is not just a scaling factor. Because when a target is moving, the attenuation changes with the transmission distance between the target and a specific sensor.

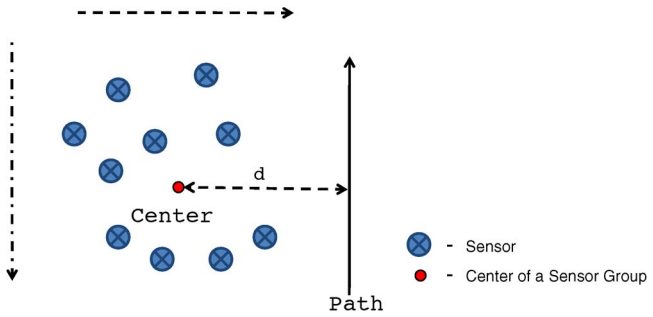

Fig. 5. Setup for experiments on signal attenuation. In the figure, the solid line and the dashed lines represent the moving paths taken by the target of interest and other targets respectively.

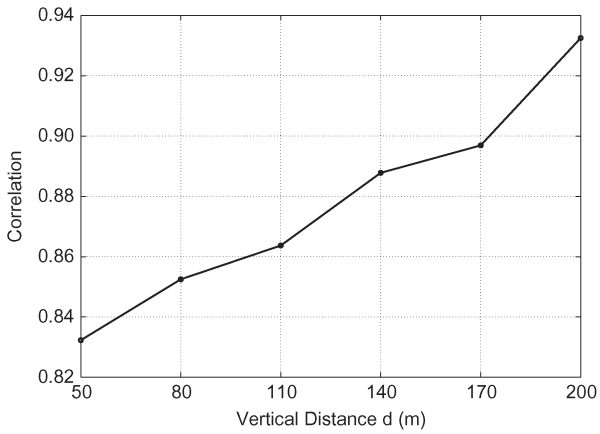

Fig. 6. Effect of attenuation.

So the difference can not be absorbed by the mixing matrix. The consequences of the difference are: (a) Noise segments can be generated during separation because of the difference (b) Separated individual signals are less correlated with original individual signals. Clustering step, selection step and voting step are designed with consideration of these consequences.

To show the effect of signal attenuation on the separation performance, we perform a simple experiment with moving targets. The experiment setup is as shown in Fig. 5: Ten randomly placed sensors form a sensor group. Three targets are moving in the sensing range of the sensor group. We fixed the path of two targets (in dashed line) in our experiment and increased d the vertical distance between the center of the sensor group and the path taken by the target of interest. To focus our investigation on the effect of signal attenuation on the separation performance, we assume there is no interference from other targets. Fig. 6 shows the maximum correlation between separated signals and the actual individual signal from the target of interest. As we can observe that when the vertical distance increases, the correlation with original individual signal is higher. So the separation performance is better when the vertical distance increases. The reason is: When the vertical distance increases, the attenuation changes less. In turn, attenuated individual signals received by different sensors are less different from each other so that better separation performance can be achieved. From this experiment, we can also infer the effect of the sensor density. When more sensors are deployed in a field, it is more likely to have a sensor group both covering the path of interest and being distant from

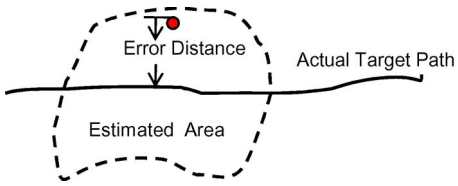

Fig. 7. Error Distance. The area within the dashed line is the estimated area and the error distance between a dot within estimated area and the actual target path is shown in the figure.

the target at the same time. So a higher sensor density can lead to better separation performance.

B. Tracking Resolution

We analyze the tracking resolution of the algorithm in this section. The purpose of the analysis is to estimate achievable performance of the proposed tracking algorithm. We focus on the intersection step in the analysis.

First, we define *error distance* as follows:

Definition 5.1: The error distance between a point in one intersection area and the path of interest is the minimal distance between the point and any point on the path.

Mathematically, the error distance d_{err} between a point (x, y) in an estimated area A and an actual target path P is defined as follows:

$$d_{err}(x, y) = \min_{(x_p, y_p) \in P} |(x, y) - (x_p, y_p)|_2 \tag{6}$$

where (x_p, y_p) represents a point on the actual target path P and $|\ |_2$ denotes the L_2-norm.

Tracking resolution is defined based on the error distance definition as follows:

Definition 5.2: Tracking resolution is defined as the average of error distance between all the points inside an intersection area and a path fragment of interest.

Mathematically, the tracking resolution TR is defined as follows:

$$TR = \frac{\int_{(x,y)\in A} d_{err}(x, y) dx\, dy}{\int_{(x,y)\in A} dx\, dy} \tag{7}$$

As shown in Fig. 7, error distance d_{err} is the minimum distance between the point inside estimated intersection area (represented with dot) and points on the path fragment of interest. Tracking resolution is the average error distance of all the points inside an estimated intersection area.

We focus on linear path fragments in theoretical analysis for the following reasons: (a) Any path can be formed with linear fragments. (b) In practice, the size of signal segment used in the proposed algorithm is small so that estimated path fragments are close to linear. To simplify the analysis of the tracking resolution, we assume the path fragment of interest fits inside the intersection area and it is perpendicular to the line joining centers of two sensor groups. We assume the sensors are uniformly distributed over the field. So sensor groups are also uniformly distributed over the field.

We assume N sensors are deployed in a field of size a meter by b meter and the sensing range of each sensor is R. Both the average tracking resolution and the finest tracking resolution are analyzed below.

1) Finest Tracking Resolution: The finest tracking resolution is defined as the achievable minimal mean error distance. We assume sensor groups are located within circles of radius r on average. So we have

$$\text{Sensor Density} = \frac{N}{a \times b} = \frac{n_{\text{group}}}{\pi r^2}$$

where n_{group} is the number of sensors in each sensor group. Thus the average radius r is

$$r = \sqrt{\frac{n_{\text{group}} ab}{\pi N}} \tag{8}$$

Theorem 5.3: The finest tracking resolution of tracking a linear path fragment of length l is $((R+r)^2/(4l)) \sin^{-1}(l/(2(R+r))) - (1/8)\sqrt{(R+r)^2 - (l/2)^2}$.

The proof of Theorem 5.3 can be found in Appendix B.

Corollary 5.4: When the finest tracking resolution is achieved, the distance between the two neighboring sensor blocks is $2\sqrt{(R+r)^2 - (l/2)^2}$.

Corollary 5.4 can be easily proven from Theorem 5.3.

2) Average Tracking Resolution: The average tracking resolution predicts the average tracking accuracy achievable by the proposed tracking algorithm. It is the mean error distance averaged over all the possible cases.

Theorem 5.5: The average tracking resolution of tracking a linear path fragment of length l is $((R+r)^2/(4l^2)) \sin^{-1}(l/(2(R+r)))((R+r) - 2\sqrt{(R+r)^2 - (l/2)^2}) + (3(R+r)^2/(16l)) - (l/16)$.

The proof of Theorem 5.5 can be found in Appendix C.

VI. Performance Evaluation

We further evaluate the performance of the proposed tracking algorithm with extensive simulations. We assume acoustic sensors are deployed in the field of interest for tracking purpose. To avoid repetition and save space, we leave empirical experiments and experiments on sensor density in [12].

A. Experiment Setup

In the following experiments, the simulated field is a 1600 m $\times$ 1600 m square area. Sensors are randomly deployed in the field. The movement of targets is restricted to a 1000 m $\times$ 1000 m *center area* to eliminate boundary effects. The signals used for tracking are real bird signals downloaded from the website of Florida Museum of Natural History [13]. In our simulation experiments, we use FastICA [14] algorithm for signal separation. FastICA is an efficient and popular algorithm for independent component analysis in terms of accuracy and low computational complexity. The attenuation of sound signals is according to atmospheric sound absorption model [15], [16]. The simulations are performed in Matlab. Following parameters are used in our experiments if not specified: (1) The sensing range of sensors is 250 m. (2) Paths followed by targets are generated randomly. (3) The number of sensors in each sensor group n_{group} is 10. (4) The number of moving targets in the

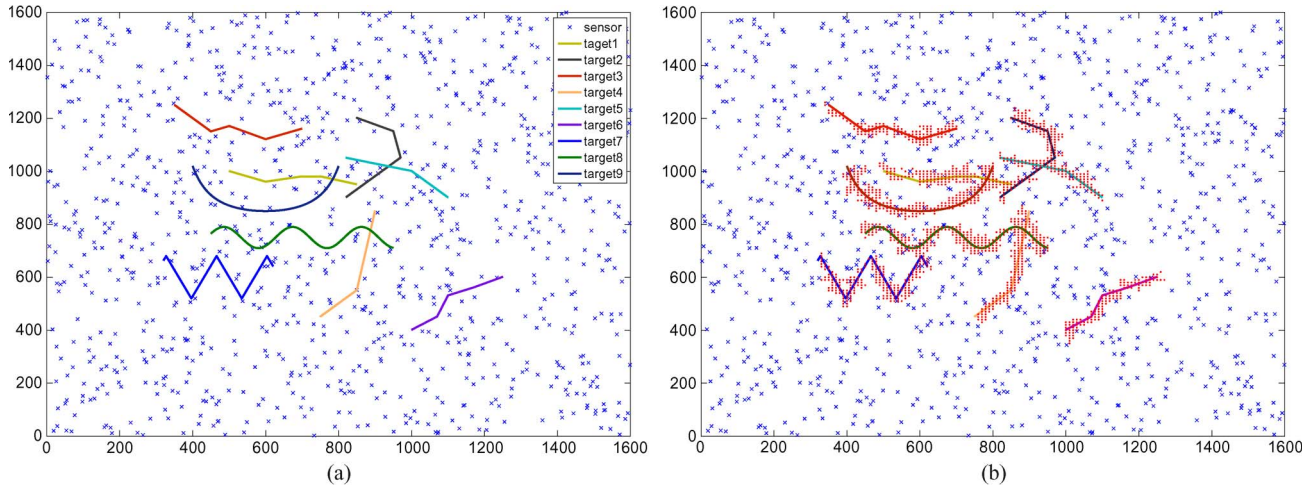

Fig. 8. An Example (Sensor Density: 1000 Sensors, $n_{\text{group}} = 10$, $l_{\text{seg}} = 100$, $l_{\text{step}} = 10$). (a) Experiment setup; (b) tracking result.

field is 10. (5) The segment length is 100 samples and the step size is 10 samples. (6) Targets are moving at a speed below 0.15 meter per sample interval. (7) In most of the experiments presented below, we set the sensor density to be 400, 700, and 1000 sensors to show tracking performance under different sensor densities.

B. Performance Metrics

In general, any tracking algorithms based on intersecting sensing ranges will output intersection areas. As described in Sections IV-E and IV-F, the estimated paths output by the target-tracking algorithm is essentially concatenated intersection areas. To evaluate the performance according to the concatenated intersection areas, we quantize the whole area using 5 m $\times$ 5 m tiles. One intersection area is represented by a set of points inside the area, each point representing the corner of the corresponding tile. Two metrics are used to evaluate the area: One is the *mean error distance*. It is based on the error distance defined in Section V-B. The mean error distance is the mean of the error distance between all points inside concatenated intersection areas and the actual path taken by a target. The other is the *standard deviation of the error distance* between the points inside the concatenated intersection areas and the actual path taken by a target. The first one measures accuracy of the tracking algorithm and the second measures precision of the tracking algorithm. If we cast the evaluation of the estimation algorithm in terms of evaluating a statistical estimator, the accuracy corresponds to the bias of the estimator and the precision corresponds to the variance of the estimator.

The step size can affect both tracking performance and computational complexity. A big step size can reduce computation time with the cost of having gaps between concatenated intersection areas. We use *percentage of coverage* to measure the continuity in estimated paths. It is equal to one minus the ratio between the sum of distance between neighboring intersection areas and the length of the actual path. The distance between two intersection areas is defined as in Section IV-F: It is the distance between two closest points in each intersection area. If two intersection areas have overlap, the distance is zero.

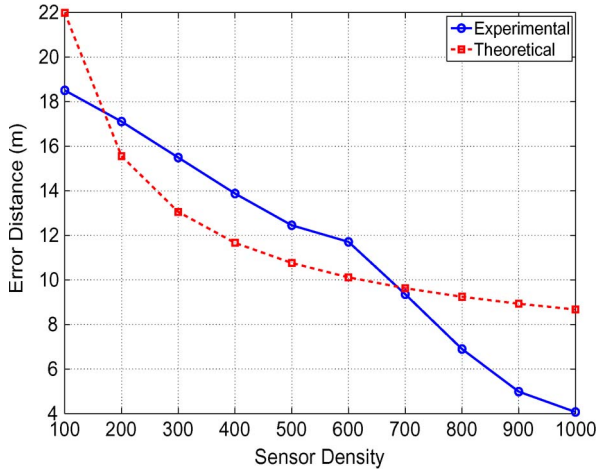

Fig. 9. Comparison between experimental results and average tracking resolution.

C. A Typical Example

An example of typical results of the proposed tracking algorithm is shown in Fig. 8. The paths taken by these targets are shown in Fig. 8(a). The sensor density is 1000 sensors in the field. We include a zigzag[5] path in this example since the zigzag path is one kind of path with high frequency variation. Fig. 8(b) shows paths estimated by our algorithm. The estimated paths are drawn in red dots. We can observe from Fig. 8 that the proposed tracking algorithm can track targets including targets following paths with high frequency variations, accurately and precisely.

D. Comparison Between Average Tracking Resolution and Experimental Results

We compared average tracking resolution derived in Section V with experimental results in this set of experiments. The results are shown in Fig. 9. For fair comparison, we fix targets' moving speed at 0.03 meter per sample in this set

[5]More experiments of the zigzag path with high frequency variation are available in Section VI-J.

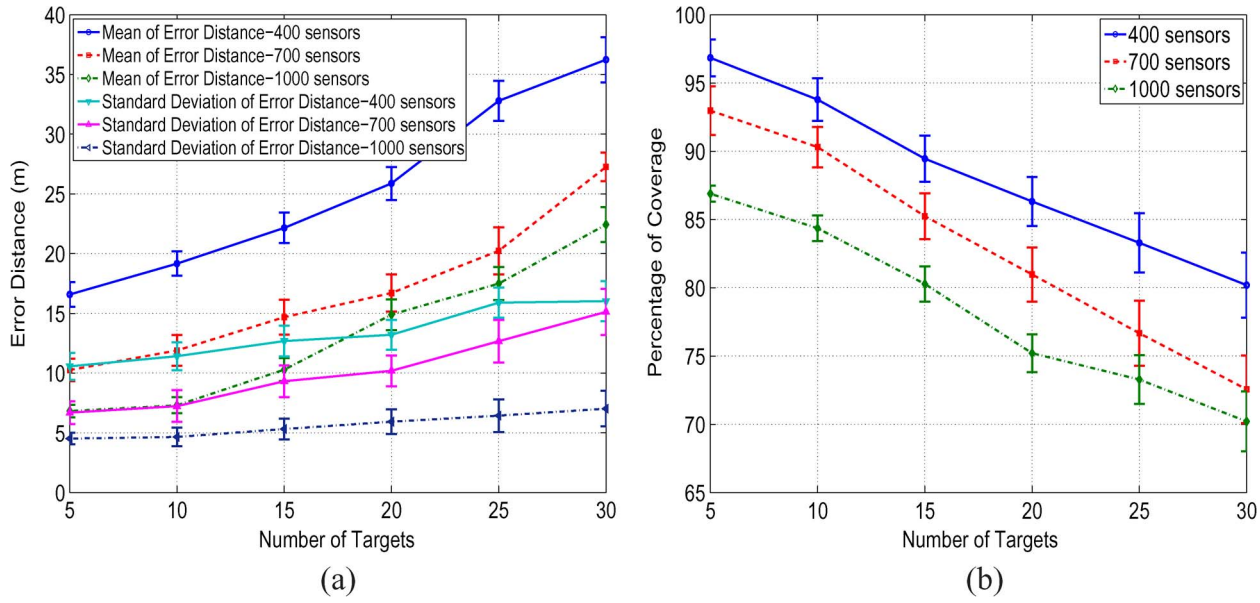

Fig. 10. Tracking performance for different number of targets: with 95 percent confidence interval. (a) Error distance; (b) percentage of coverage.

of experiments. We can observe the experimental curve is close to the curve of average tracking resolution. The experimental results are in the same order of the average tracking resolution. When the sensor density is larger than 1000, the difference between the two curves becomes smaller because (a) the error distance decreases when the sensor density increases for both curves and (b) the difference between these two curves is less than 9 meters when sensor density is larger than 1000. More experiment results on sensor density can be found in [12].

E. Number of Targets

In this set of experiments, we vary the number of targets moving in the field. The results are shown in Fig. 10. From Fig. 10(a), we can observe: (a) When the field is crowded with targets, our algorithm can still track targets with reasonable accuracy and precision. (b) The error distance increases when the number of targets increases. It is because the separation step can not perfectly separate out all the signals when the number of moving targets increases. As shown in Fig. 10(b), the percentage of coverage decreases when the number of targets increases. The decrease is caused by the decrease in separation performance so that path fragments estimated for different time slots are less consistently covering the actual paths.

F. Moving Speed

In this set of experiments, we investigate the effect of the moving speed on tracking performance. Targets in this set of experiments are moving with different speed. From experiment results shown in Fig. 11, we can observe that the error distance increases when the moving speed increases. It is because that the speed increase can lead to decrease of the separation performance and less number of sensor groups sensing enough signal for tracking.

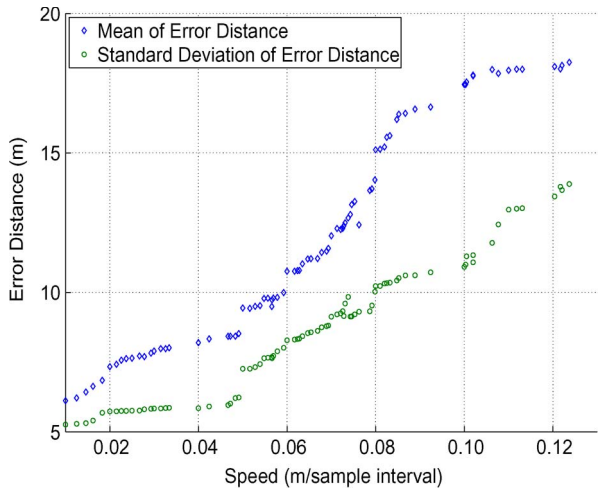

Fig. 11. Scatter plot of tracking performance vs. moving speed.

G. Segment Length (l_{seg})

This set of experiments focus on the length of signal segments used in tracking algorithm. In this set of experiments, we fix the step size at 10 samples and vary the segment length. Since the tracking algorithm processes signals in the unit of segments, the segment length is a critical parameter for the algorithm. The experiment results are shown in Fig. 12. The results in Fig. 12(a) indicate: The error distance increases when the segment length increases. It is because of less number of sensor groups which can "hear" targets for the whole path fragment in their sensing ranges. The decrease in the number of sensor groups also causes the decrease in percentage of coverage as shown in Fig. 12(b).

H. Step Size (l_{step})

In this set of experiments, we fix the segment length at 100 samples and vary the step size. As shown in Fig. 13(a), the error distance increases with the step size. This is because

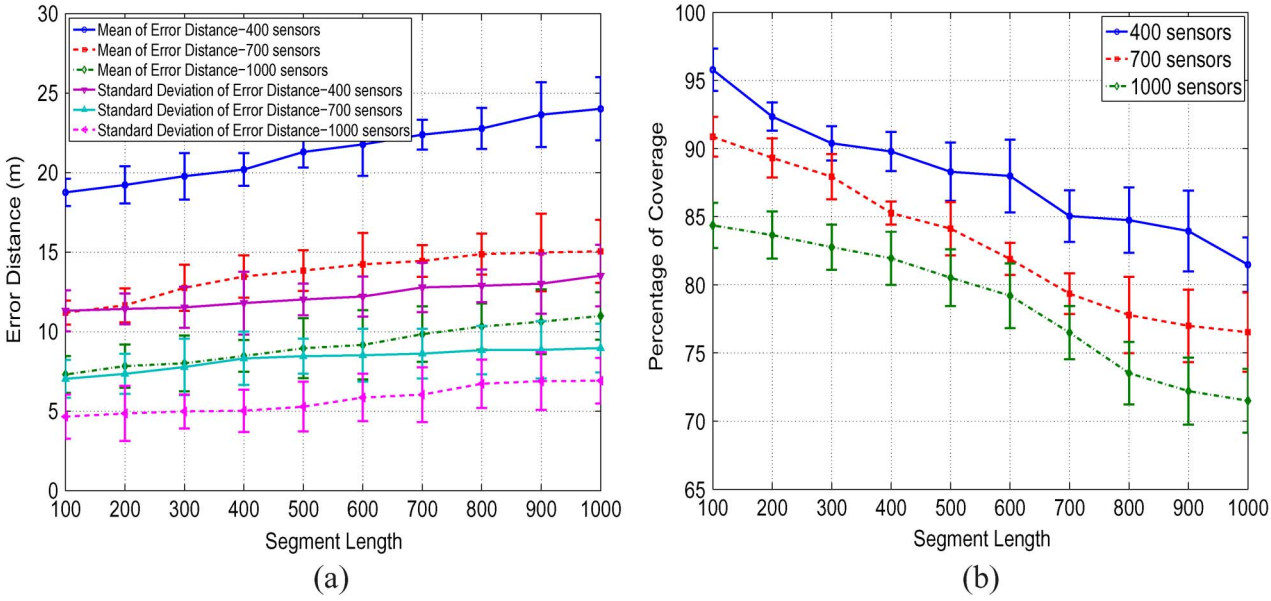

Fig. 12. Effect of signal segment length (l_{seg}) on tracking performance: with 95 percent confidence interval. (a) Error distance; (b) percentage of coverage.

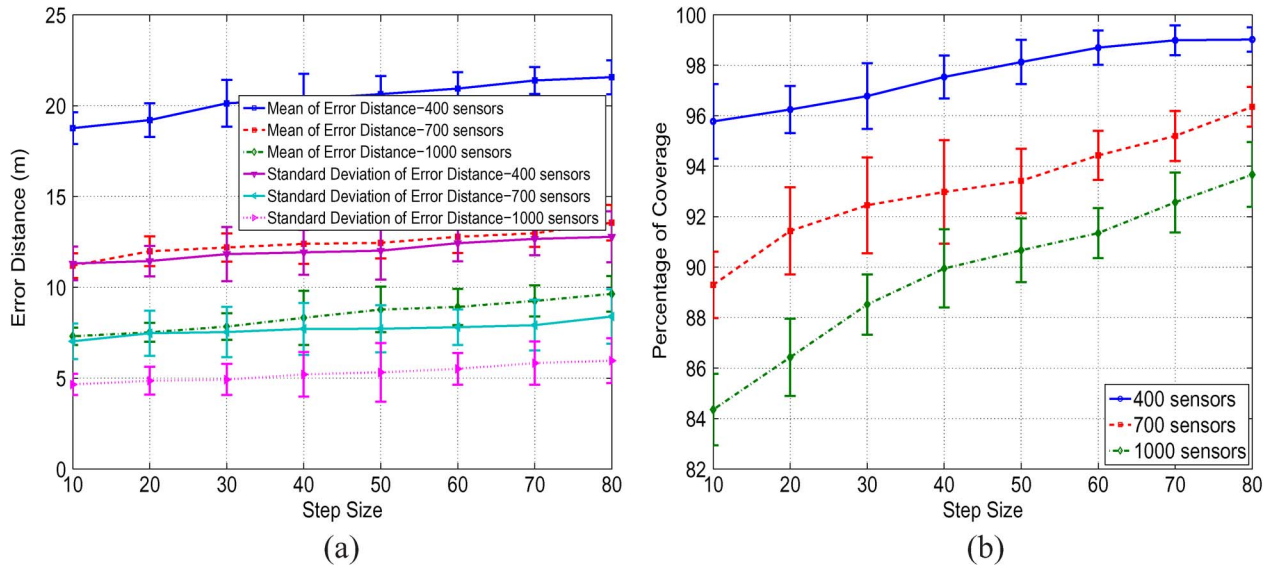

Fig. 13. Effect of step size (l_{step}) on tracking performance: with 95 percent confidence interval. (a) Error distance; (b) percentage of coverage.

for a certain segment length, a larger step size reduces the length of common part of two successive time slots. In turn, the link correlation becomes less reliable. When the step size increases, the percentage of coverage also increases. The less reliable link correlation, caused by a larger step size, lead to the larger intersection areas. The larger intersection areas increases the percentage of coverage.

I. Effect of Number of Sensors in Sensor Groups

In this subsection, we describe our experiments on the parameter n_{group}, i.e., the number of sensors in each sensor group. The results are shown in Fig. 14. As shown in the figure, the error distance is larger when n_{group} is too small or too large. When n_{group} is small, the number of targets can be larger than the number of sensors in a sensor group. Generally BSS algorithms perform better when the number of observations is larger than the number of individual signals. So more sensors in a sensor group can lead to better separation performance. But when the number of sensors in sensor group increases, the sensing range also increases. This lead to larger intersection areas when intersecting these larger sensing areas in the intersection step.

J. Paths With High-Frequency Variations

In this set of experiments, we experiment on the performance of tracking targets following paths with high-frequency variations. In the experiments, we focus on paths between two points A and B with distance of 300 m from each other as shown in Fig. 15. Paths between these two points are zigzag paths of different periods. The width of the path is 100 m and we vary zigzag period in our experiments. From the results shown in Fig. 16, we can observe the tracking algorithm can track targets following zigzag paths accurately. We believe that the slight

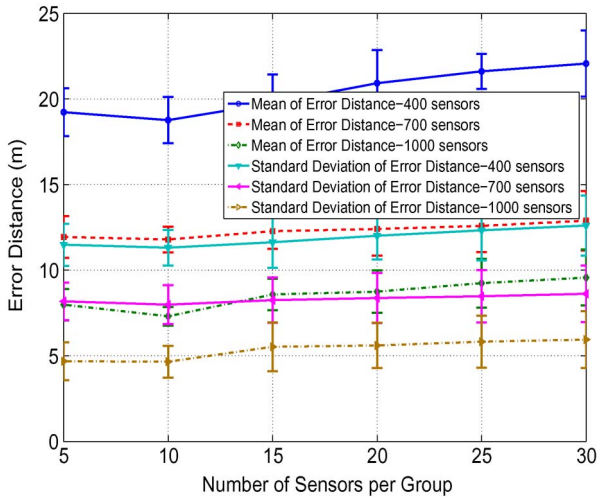

Fig. 14. Effect of number of sensors in sensor groups: with 95 percent confidence interval.

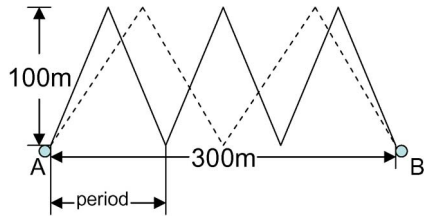

Fig. 15. Example of zigzag paths.

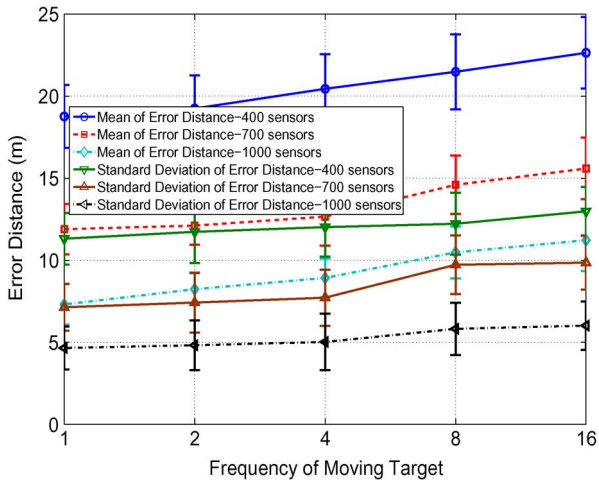

Fig. 16. Path with high frequency variation: with 95 percent confidence interval.

increase of error distance with the number of zigzag periods is because of higher speed required to finish longer paths. This experiments demonstrate the benefit of applying BSS algorithms in tracking targets. It enables tracking algorithms to have richer information for target-tracking. So the proposed algorithm can successfully track targets following paths with high-frequency variations.

VII. Related Work

In this section, we first review related work on target tracking via wireless sensor networks with focus on tracking schemes based on Received Signal Strength (RSS). Then we review primary user localization schemes in the context of cognitive radio networks. Finally, we discuss Blind Source Separation (BSS) algorithms.

Most approaches proposed to track targets or detect location are based on characteristics of physical signals such as Angle of Arrival (AOA),Time of Arrival (TOA),Time Difference of Arrival (TDOA), and Received Signal Strength (RSS). The RSS is widely used in tracking targets with wireless sensor networks [17]–[24]. Our approach is based on RSS.

Most of the previous researches focus on tracking a single target [25]–[27] or assume targets are distinguishable [28]. A string of researches on tracking targets with wireless sensor networks are based on binary proximity sensors which can only report whether there are targets within sensing areas. The initial work [25]–[27] on binary proximity sensors focuses on tracking single target. Singh *et al.* [29] extended the approach to track multiple indistinguishable targets by applying particle filtering algorithms. Approaches based on binary proximity sensors have two obvious advantages: (a) The sensors are very simple since they only report binary information. (b) The approaches are robust since interference from other targets are essentially filtered out by an equivalent low-pass filter [27]. The disadvantage of the approaches based on the simple devices is the loss of information that is helpful to accurately track targets due to the filtering effect. So, approaches based on binary proximity sensors can not track targets in paths with high-frequency variations [27]. We propose a general approach to track multiple indistinguishable targets. The approach is based on blind source separation algorithms, which can recover individual signals from aggregate signals. Since individual signals can be fully recovered, our approach can track targets following paths with high-frequency variations.

Cognitive radio is considered as a novel approach for efficient utilization of the radio spectrum [30]. The Emerging cognitive radio applications rang from smart grid, public safety and broadband cellular, to medical applications [31]. In cognitive sensor networks (CSN), location information of the primary user (PU) is important for secondary users (SU) in order to efficiently utilize spatial spectrum while protecting primary communication. Some schemes have been proposed on primary user localization in the context of CSN [32]–[34]. The proposed schemes [32]–[34] differ from our work in two aspects. First of all, they usually assume that the secondary network is an infrastructure-based network, such as an IEEE 802.11 WRAN, in which each cell consists of a central processing point, e.g., a base station, to fuse the information to estimate the position of the PU. Secondly, the proposed schemes either focus on the tracking of only one mobile PUs location after its detection [34] or assume the primary users are almost stationary [32], [33].

According to our knowledge, we are the first to propose tracking algorithms with wireless sensor networks based on the BSS algorithms. The proposed tracking algorithms use the general separation algorithm, the FastICA [14] algorithm, to recover individual signals. We propose algorithms to remove noise and artifacts created by BSS algorithms so that the tracking algorithm can both accurately and precisely track multiple indistinguishable targets.

VIII. Discussion and Future Work

In this section we discuss the algorithm complexity and outline our future work.

The complexity of the algorithm is largely determined by the step size shown in Fig. 3. The number of separations performed by the algorithm is in the order of $O((L/l_{\text{step}}) \times N_{\text{grps}})$, where L is the total number of samples in one aggregate signal, l_{step} is the step size and N_{grps} is the number of sensor groups. A larger step size can reduce the number of separations performed by the algorithm. The cost will be slight degradation of tracking performance as shown in Fig. 13(a).

We plan to investigate the effect of time synchronization on the tracking performance. The clock differences among sensor nodes can possibly reduce the correlation used in the clustering step and the selection step. Existing time synchronization protocols such as [35]–[37] can be used to synchronize the clocks in sensor nodes. We can also tolerate the clock differences by adjusting sampling of wireless sensors. The adjusting can be changing the length of sampling intervals or aggregating successive samples into one sample so that the correlation can still be largely kept for the processing in the proposed algorithm.

One interesting research topic for our future work is applying data compression algorithms to reduce the data volume as redundancy exists spatially in signals reported by neighboring sensors and temporally in a signal reported by the same sensor node. We plan to apply existing algorithms such as ESPIHT [4] to verify the benefit of the data compression.

We plan to investigate the effect of signal interference, inaccurate sensing, and sensor node failures. The effect can be modeled as noise in the aggregate signals received by the sensors. The noise can be largely removed by the clustering step and the selection step. But it is still interesting to see how the tracking algorithm performs in extreme conditions such as a large percentage of node failures.

IX. Conclusion

We propose an approach to track non-cooperative targets using wireless sensor networks. The approach is based on blind source separation (BSS) algorithms. By applying BSS algorithms on aggregate signals collected from sensors, we can recover individual signals from targets for tracking. The proposed tracking algorithm fully utilizes both spatial and temporal information available for tracking. We evaluate the proposed tracking algorithm both experimentally and theoretically. Our experiment results show that the tracking algorithm can track targets both accurately and precisely. Because of richer information made available by BSS algorithms, the proposed algorithm can also track paths with high-frequency variations.

Appendix A
Pseudo Code of the Voting Step

The pseudo code of the voting step is as shown in Algorithm 1. To help readers understand the algorithm, we describe the sketch of the algorithm as follows: First the step selects the "best" path segments according to $dcur_{k,u}$ and $dpre_{k,u}$ as described in Section IV-F. Then the step decides whether a selected path segment belongs to a path based on the distance between the selected path segment and the path segment decided for the previous time slot.

Algorithm 1: Voting Step

input : K - number of estimated segments in each time slot, v - number of time slots available, $area^u_{i,j}$ - the ith estimated segment among all the estimated segments in the jth time slot based on CTS_u;
output: $epath_l$ - Estimated path of the lth moving target.

for $u \leftarrow 1$ **to** $v - n_{slot} + 1$ **do**
  for $k \leftarrow 1$ **to** K **do**
    $dcur_{k,u} = \sum_{j=u}^{u+n_{slot}-2} d_{area}(area^u_{k,u}, area^u_{k,j+1})$;
      /* $dcur_{k,u}$ is sum of distance between estimated segments in the uth time slot and other time slots estimated in the same CTS_u */
    if $u > 1$ && $u < n_{slot}$ **then**
      /* Boundary Case */
      $dpre_{k,u} = \sum_{m=1}^{u-1} min(d_{area}(area^u_{k,u}, area^{u-m}_{1,u}), d_{area}(area^u_{k,u}, area^{u-m}_{2,u}), \cdots, d_{area}(area^u_{k,u}, area^{u-m}_{K,u}))$;
      /* $dpre_{k,u}$ is sum of minimum distance between estimated segments in the uth time slot in current CTS and the uth time slot in all previous CTS */
    else
      $dpre_{k,u} = \sum_{m=1}^{n_{slot}-1} min(d_{area}(area^u_{k,u}, area^{u-m}_{1,u}), d_{area}(area^u_{k,u}, area^{u-m}_{2,u}), \cdots, d_{area}(area^u_{k,u}, area^{u-m}_{K,u}))$;
    end
    if $dcur_{k,u} == 0$ && $dpre_{k,u} == 0$ **then**
      $Pseg_{k,u} = area^u_{k,u}$; /* $Pseg_{k,u}$, denote the kth estimated segment in the uth time slot */
    else
      $Pseg_{k,u} = -1$; /* -1, means not selected */;
    end
    if $u == v - n_{slot} + 1$ **then**
      for $i \leftarrow u$ **to** $v - 1$ **do**
        if $d_{area}(area^u_{k,u}, area^u_{k,i+1}) == 0$ **then**
          $Pseg_{k,i+1} = area^u_{k,i+1}$;
        else
          $Pseg_{k,i+1} = -1$;
        end
      end
    end
  end
end
foreach *target* $l \leftarrow 1$ **to** K **do**
  for $j \leftarrow 2$ **to** v **do**
    if $j == 2$ **then**
      if $min(d_{area}(Pseg_{l,j}, Pseg_{1,j+1}), d_{area}(Pseg_{l,j}, Pseg_{2,j+1}), \cdots, d_{area}(Pseg_{l,j}, Pseg_{K,j+1})) == 0$ **then**
        $Pseg_{l,j} \in epath_l$;
        $Pseg_{z_{j+1},j+1} \in epath_l$; /* Without loss of generality, it is assumed that $d_{area}(Pseg_{l,j}, Pseg_{z_{j+1},j+1}) = 0$. If more than two segments have zero distance with $Pseg_{l,j}$, the tiebreaker is the index x in $Pseg_{x,j+1}$ */
      else
        $Pseg_{l,j} \in epath_l$;
      end
    else
      if $min(d_{area}(Pseg_{z_{j-1},j-1}, Pseg_{1,j}), d_{area}(Pseg_{z_{j-1},j-1}, Pseg_{2,j}), \cdots, d_{area}(Pseg_{z_{j-1},j-1}, Pseg_{K,j})) == 0$ **then**
        $Pseg_{z_j,j} \in epath_l$; /* Without loss of generality, it is assumed that $d_{area}(Pseg_{z_{j-1},j-1}, Pseg_{z_j,j}) = 0$ */
      end
    end
    if $Pseg_{z_j,j} \in epath_l$ **then**
      continue;
    else
      find the last determined segment in $Pseg_{z_x,x}$ in $epath_l$;
      find $min(d_{area}(Pseg_{z_x,x}, Pseg_{1,j}), d_{area}(Pseg_{z_x,x}, Pseg_{2,j}), \cdots, d_{area}(Pseg_{z_x,x}, Pseg_{K,j}))$;
      $Pseg_{z_j,j} \in epath_l$; /* with out loss of generality, it is assumed that $d_{area}(Pseg_{z_x,x}, Pseg_{z_j,j}) = 0$ */
    end
  end
end

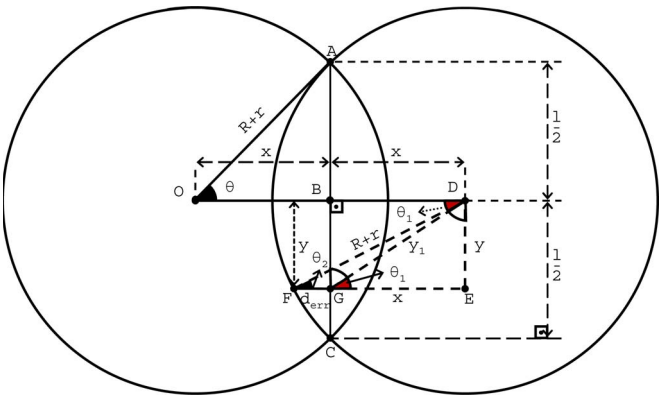

Fig. 17. Finest tracking resolution.

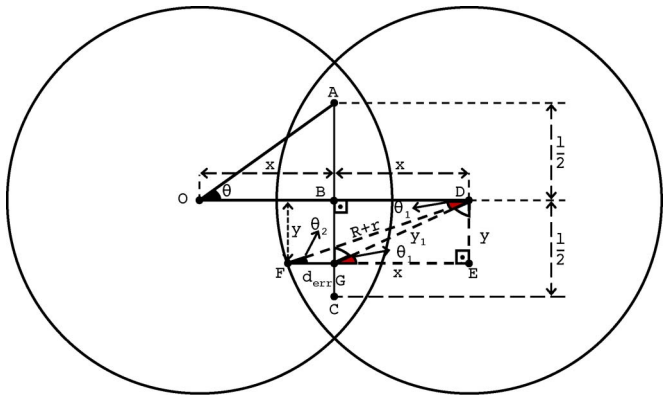

Fig. 18. Average tracking resolution (difference from Fig. 17: points A and C are not on the circles).

APPENDIX B
PROOF OF THEOREM 5.3

Proof: The finest tracking resolution is achieved when the path segment of interest fits exactly into the intersection area of two sensing ranges as shown in Fig. 17. It can be proven otherwise the tracking resolution becomes worse. In Fig. 17, line segment AC is the linear path segment of length l. The corresponding estimated path segment covers the overlap of the two neighboring sensing ranges. So the path segment of interest is perpendicular to the line joining centers of sensor groups. The distance d_{err} is the distance between the sample point on the path denoted with G and the point on the perimeter of the sensing range denoted with F. Since d_{err} is the shortest distance from F to any points on the path segment, d_{err} is also the error distance between point F and the path segment. Suppose in Fig. 17 the distance between centers of two neighbor sensor groups is $2x$. The value of x can be derived as follows. $\triangle OAB$ is a right angle triangle, $OA = R + r$ (the sensing radius of sensor group), and $AB = l/2$.

From $\triangle OAB$,

$$x = \sqrt{(R+r)^2 - \left(\frac{l}{2}\right)^2}. \tag{9}$$

Thus the distance between neighbor sensor groups is $2x$, i.e., $2\sqrt{(R+r)^2 - (l/2)^2}$.

The error distance d_{err} can be derived as follows: From $\triangle DGE$ as shown in Fig. 17,

$$\tan\theta_1 = \frac{y}{x},$$
$$\theta_1 = \tan^{-1}\frac{y}{x}, \tag{10}$$
$$x = \frac{y}{\tan\theta_1}. \tag{11}$$

Now from $\triangle FDE$, $FD = R + r$ (the sensing radius of sensor group). We denote the distance between the point B and G with y. From $\triangle FDE$,

$$\theta_2 = \sin^{-1}\left(\frac{y}{R+r}\right), \tag{12}$$

and

$$\tan\theta_2 = \frac{y}{d_{err}+x}, \quad d_{err} = \frac{y}{\tan\theta_2} - x, \tag{13}$$

where x is from Equation (11) and θ_2 is from Equation (12). Since

$$d_{err} = \frac{y}{\tan\theta_2} - \frac{y}{\tan\theta_1}, \tag{14}$$

we can further simplify the above equation by substituting θ_1 and θ_2 values derived in Equation (10) and Equation (12) respectively. So d_{err} can be derived as follows:

$$d_{err} = (R+r)\cos(\theta_2) - \sqrt{(R+r)^2 - \left(\frac{l}{2}\right)^2}. \tag{15}$$

For all the points on the line segment FG, the average error distance is $d_{err}/2$. Integral is used to calculate average of error distance for all the points within the intersection area. Thus the finest tracking resolution is $(1/2)\int_0^{l/2}\{((R+r)\cos(\theta_2) - \sqrt{(R+r)^2 - (l/2)^2})/l\}\,dy$ where $\theta_2 = \sin^{-1}(y/(R+r))$.

After further simplification the finest tracking resolution becomes $((R+r)^2/(4l))\sin^{-1}(l/(2(R+r))) - (1/8)\times\sqrt{(R+r)^2 - (l/2)^2}$. ■

APPENDIX C
PROOF OF THEOREM 5.5

Proof: From $\triangle FDE$ as shown in Fig. 18,

$$\cos\theta_2 = \frac{y}{R+r}, \quad \theta_2 = \cos^{-1}\left(\frac{y}{R+r}\right). \tag{16}$$

We derived the error distance d_{err} in Appendix B.

$$d_{err} = \frac{y}{\tan\theta_2} - \frac{y}{\tan\theta_1}, \tag{17}$$

we can further simplify the above equation by substituting θ_1 and θ_2 values derived in Equation (10) and Equation (16) respectively. Then d_{err} can be derived as follows:

$$d_{err} = (R+r)\cos\theta_2 - x \tag{18}$$

where $\theta_2 = \sin^{-1}(y/(r+R))$. So the mean error distance is $(1/2)\int_0^{l/2}\{(R+r)\cos\theta_2 - x\}\,dy$.

From Corollary 5.4, we know the distance between the centers of two sensor groups when the finest tracking resolution is achieved. The worst-case tracking resolution is achieved when the distance between the centers of two sensor groups is $R + r$. The average tracking resolution can be derived by integral of mean error distance over possible distance between the centers of two sensor groups. So the average tracking resolution is $(1/2)\int_Z^{R+r}\int_0^{l/2}\{((R+r)\cos\theta_2 - x)/l^2\}\,dy\,dx$ where $Z = 2\times\sqrt{(R+r)^2-(l/2)^2}$ and $\theta_2 = \sin^{-1}(y/(r+R))$.

After further simplification the average tracking resolution is $((R+r)^2/(4l^2))\sin^{-1}(l/(2(R+r)))((R+r) - 2\sqrt{(R+r)^2-(l/2)^2}) + (3(R+r)^2/16l) - (l/16)$. ■

ACKNOWLEDGMENT

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] L. Tong, R. Liu, V. Soon, and Y.-F. Huang, "Indeterminacy and identifiability of blind identification," *IEEE Trans. Circuits Syst.*, vol. 38, no. 5, pp. 499–509, May 1991.

[2] C. Savarese, J. Rabaey, and K. Langendoen, "Robust positioning algorithms for distributed ad-hoc wireless sensor networks," in *Proc. Gen. Track Annu. Conf. USENIX ATEC*, 2002, pp. 317–327.

[3] C. Tang and C. S. Raghavendra, *Compression Techniques for Wireless Sensor Networks*. Norwell, MA, USA: Kluwer, 2004, pp. 207–231.

[4] J. Cardoso, "Blind signal separation: Statistical principles," *Proc. IEEE*, vol. 86, no. 10, pp. 2009–2025, Oct. 1998.

[5] P. Comon, "Independent component analysis, a new concept?" *Signal Process.*, vol. 36, no. 3, pp. 287–314, 1994.

[6] A. Hyvrinen and E. Oja, "A fast fixed-point algorithm for independent component analysis," *Neural Comput.*, vol. 9, pp. 1483–1492, 1997.

[7] M. Gaeta and J.-L. Lacoume, "Source separation without a priori knowledge: The maximum likelihood solution," in *Proc. EUSIPCO*, Barcelona, Spain, 1990, pp. 621–624.

[8] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*., 2nd ed. Hoboken, NJ, USA: Wiley, 2001.

[9] Y. Baryshnikov and R. Ghrist, "Target enumeration via Euler characteristic integrals," *SIAM J. Appl. Math*, vol. 70, no. 3, pp. 825–844, 2009.

[10] Q. Fang, F. Zhao, and L. Guibas, "Lightweight sensing and communication protocols for target enumeration and aggregation," in *Proc. 4th ACM Int. Symp. MobiHoc Netw. Comput.*, 2003, pp. 165–176.

[11] S. R. Blatt, "Target Resolution in Distributed Sensor Systems," NASA STI, Hampton, VA, USA, Recon Tech. Rep. N, 3:15 776, Oct. 2001.

[12] Y. Zhu, A. Vikram, and H. Fu, "On tracking multiple indistinguishable targets," in *Proc. 9th IEEE Int. Conf. MASS*, Oct. 2012, pp. 1–9.

[13] J. W. Hardy, Sounds of Florida's Birds, Florida Museum of Natural History, Florida Museum of Natural History, Gainesville, FL, USA. [Online]. Available: http://www.flmnh.ufl.edu/birds/sounds.htm

[14] C. W. Hesse and C. J. James, "The fastica algorithm with spatial constraints," *IEEE Signal Process. Lett.*, vol. 12, no. 11, pp. 792–795, Nov. 2005.

[15] Kinsler *et al.*, *Fundamentals of Acoustics*. New York, NJ, USA: Wiley, 2000.

[16] H. E. Bass, L. C. Sutherland, A. J. Zuckerwar, D. T. Blackstock, and D. M. Hester, "Atmospheric absorption of sound: Further developments," *J. Acoust. Soc. Am.*, vol. 97, no. 1, pp. 680–683, 1995. [Online]. Available: http://link.aip.org/link/?JAS/97/680/1

[17] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proc. 19th IEEE INFOCOM*, Tel Aviv, Israel, 2000, vol. 2, pp. 775–784.

[18] S.-P. Kuo, Y.-C. Tseng, F.-J. Wu, and C.-Y. Lin, "A probabilistic signal-strength-based evaluation methodology for sensor network deployment," in *Proc. 19th Int. Conf. AINA*, Taipei, Taiwan, Mar. 2005, vol. 1, pp. 319–324.

[19] P. Tarrio, A. Bernardos, and J. Casar, "An rss localization method based on parametric channel models," in *Proc. Int. Conf. SensorComm Technol. Appl.*, Oct. 2007, pp. 265–270.

[20] N. Patwari and A. O. Hero, III, "Using proximity and quantized rss for sensor localization in wireless networks," in *Proc. 2nd ACM Int. Conf. WSNA*, 2003, pp. 20–29, New York, NY, USA: ACM.

[21] R. Vaghefi, M. Gholami, and E. Strom, "Rss-based sensor localization with unknown transmit power," in *Proc. IEEE ICASSP*, May 2011, pp. 2480–2483.

[22] E. Elnahraway, X. Li, and R. P. Martin, "The limits of localization using rss," in *Proc. 2nd Int. Conf. Embedded Netw. SenSys*, 2004, pp. 283–284.

[23] J. Shirahama and T. Ohtsuki, "Rss-based localization in environments with different path loss exponent for each link," in *Proc. IEEE VTC Spring*, May 2008, pp. 1509–1513.

[24] T. Stoyanova, F. Kerasiotis, A. Prayati, and G. Papadopoulos, "Evaluation of impact factors on rss accuracy for localization and tracking applications," in *Proc. 5th ACM Int. Workshop MobiWac*, 2007, pp. 9–16.

[25] J. Aslam *et al.*, "Tracking a moving object with a binary sensor network," in *Proc. 1st Int. Conf. Embedded Netw. SenSys*, 2003, pp. 150–161.

[26] W. Kim, K. Mechitov, J.-Y. Choi, and S. Ham, "Target tracking with binary proximity sensors," in *Proc. IPSN*, Los Angeles, CA, USA, Apr. 2005, pp. 301–308.

[27] N. Shrivastava, R. M. U. Madhow, and S. Suri, "Target tracking with binary proximity sensors: Fundamental limits, minimal descriptions, algorithms," in *Proc. 4th Int. Conf. Embedded Netw. SenSys*, 2006, pp. 251–264.

[28] B. Malhotra and A. Aravind, "Path-adaptive on-site tracking in wireless sensor networks," *IEICE Trans. Inf. Syst.*, vol. E89-D, no. 2, pp. 536–545, Feb. 2006.

[29] J. Singh, U. Madhow, R. Kumar, S. Suri, and R. Cagley, "Tracking multiple targets using binary proximity sensors," in *Proc. 6th Int. Conf. IPSN*, 2007, pp. 529–538.

[30] S. Haykin, "Cognitive radio: Brain-empowered wireless communications," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 2, pp. 201–220, Feb. 2005.

[31] J. Wang, M. Ghosh, and K. Challapali, "Emerging cognitive radio applications: A survey," *IEEE Commun. Mag.*, vol. 49, no. 3, pp. 74–81, Mar. 2011.

[32] D. Gong, Z. Ma, Y. Li, W. Chen, and Z. Cao, "High order geometric range free localization in opportunistic cognitive sensor networks," in *Proc. IEEE ICC Workshops*, May 2008, pp. 139–143.

[33] S. Kandeepan, S. Reisenfeld, T. Aysal, D. Lowe, and R. Piesiewicz, "Bayesian tracking in cooperative localization for cognitive radio networks," in *Proc. IEEE 69th VTC Spring*, Apr. 2009, pp. 1–5.

[34] A. Min and K. Shin, "Robust tracking of small-scale mobile primary user in cognitive radio networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 4, pp. 778–788, Apr. 2013.

[35] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 147–163, Dec. 2002.

[36] K. Römer, "Time synchronization in ad hoc networks," in *Proc. 2nd ACM Int. Symp. MobiHoc Netw. Comput.*, 2001, pp. 173–182.

[37] M. Mock, R. Frings, E. Nett, and S. Trikaliotis, "Continuous clock synchronization in wireless real-time applications," in *Proc. 19th IEEE SRDS*, 2000, pp. 125–132.