A Monte Carlo Method for Mobile Target Counting

Dengyuan Wu*, Xiuzhen Cheng*, Dechang Chen†, Wei Cheng*, Biao Chen‡, Wei Zhao‡

*Department of Computer Science, The George Washington University, Washington, DC 20052, USA
†Division of Epidemiology and Biostatistics, Uniformed University of the Health Sciences, Betheda, MD 20814 USA
‡Department of Computer Information Science, University of Macau, Macau, P. R. China.
Email: {andrewwu,cheng,wcheng}@gwu.edu, dchen@usuhs.mil,{bchen1999,zhao8686}@gmail.com

Abstract—This paper addresses the problem of target counting based on the Monte Carlo simulation. We rely on an Accept-Reject process to guide the placement of virtual targets in a virtual sensor field, which has exactly the same sensor layout as the real one. The objective of this construction is to generate a virtual target energy landscape whose shape is close enough to an energy landscape estimated from the real sensor readings. Based on the number of virtual targets placed on the virtual field and the total virtual and real target energy volumes, the number of real targets can be estimated. We consider both single-epoch and multi-epoch sensor readings, and our theoretical analysis indicates that our approach yields an estimated target count that approximately converges to the true target count. Extensive comparison based simulation study has been performed and the results verify the effectiveness of our target counting algorithms.

I. INTRODUCTION

One of the most basic and fundamental problems of wireless sensor networks is to estimate the total number of targets in a specific monitored region. In fact, counting is usually the first step in other applications such as target tracking and positioning [1]–[6]. Consider, for example, the problem of tracking a mobile group of bison in a certain part of the Yellowstone national park. Bio-ecologists may be interested in the group size, location, and mobility patterns of the bison to infer other useful information. A sensor network could be deployed for this purpose to periodically gather relevant data.

This paper focuses on the problem of mobile target counting in sensor networks. The overlapping influence of the target energies is the main challenge of estimation. Existent research [7]–[11] has made a great effort to improve the counting accuracy but most technical approaches yield relatively good performance only when the number of targets covered by each sensor is very small and the total number of targets is significantly less than that of the sensors. In other words, these approaches are only suitable for *sparse target* counting. In *dense target* deployment where the number of targets could be much larger than that of the sensors, the overlapping influence of multiple targets is quite large, therefore novel techniques have to be sought in order to improve the counting accuracy.

We resort to the Monte Carlo simulation to count the number of targets in a monitored sensor area. The Monte Carlo simulation [12], [13] is usually employed to handle problems with a high degree of uncertainty or multiple coupled degrees of freedom. It is a viable approach for target counting in sensor networks because many of the hidden parameters, such as the sharp decrease of the sensor reading resulted from a small increase of the distance between the sensor and a target, are indirectly considered in Monte Carlo simulation. In our approach, we first construct Voronoi cells [14] based on the sensor locations, and then compute an estimated target landscape based on the single-epoch sensor readings. A virtual sensor field, with exactly the same shape and same sensor layout as the original one, is utilized. Virtual targets, whose energy decay property is the same as that of the real targets, are placed in the virtual sensor field following a controlled Accept-Reject [15] procedure. The number of real targets is then computed from the energy volumes of both the real and the virtual sensor fields and the number of virtual targets. We also enhance the counting accuracy by exploring sensor readings from multiple epochs, in which the mobility of the targets results in different sensor readings from epoch to epoch. To be specific, we run the Monte Carlo simulation on each epoch to compute the corresponding virtual energy volume. All the virtual energy volumes are accumulated, and the real target count is estimated from the cumulative virtual and real energy volumes and the total number of virtual targets.

The energy landscape or volume of the original sensor field has been employed to estimate the number of targets in [7] and [8], but neither utilizes the Monte Carlo simulation to derive the true target count. As a result, none of these approaches yields a good performance in dense target counting, though the performance after a careful parameter calibration is acceptable for the sparse scenario. Compared to the state-of-the-art, in particular, the most relevant research, this paper has the following novel contributions.

1) We propose two Monte Carlo simulation based target counting algorithms, with one for single-epoch sensor readings, denoted by MCTC-s, and the other for multi-epoch sensor readings, denoted by MCTC-m, to estimate the total number of targets. An Accept-Reject procedure is devised to control the placement of the virtual targets, whose positions are represented by a

two-dimensional random variable, in the virtual sensor field such that the produced virtual energy landscape has a similar shape to that of the real sensor field when the number of virtual targets is large enough.

2) We perform an extensive simulation study to investigate the performances of MCTC-s and MCTC-m and provide comparisons with those of VCTC and Regression, two algorithms tailored from EBAM in [7] and the heuristic in [8], which estimate the real target energy volume based on Voronoi cells and a polynomial regression function, respectively. Our simulation results verify the superiority of MCTC-s and MCTC-m over VCTC and Regression for both sparse and dense target deployments. We also present an in-depth analysis on the factors that affect the counting accuracy.

3) We conduct a theoretical performance analysis to argue that MCTC-s and MCTC-m each approximately converges to the real target count.

The rest of the paper is organized as follows. In Section II, we review the most related research. Our network model and other background information are introduced in Section III. Section IV proposes MCTC-s, the target counting algorithm based on single-epoch sensor readings. MCTC-m, the algorithm based on multi-epoch sensor readings is presented and analyzed in Section V. Our comparison based simulation results are reported in Section VI and this paper is concluded with a discussion on future research in Section VII.

II. RELATED WORK

Prior efforts on target counting in sensor networks mainly lie in four directions: (1) binary-sensing based approaches [16]–[18] count the number of targets by assuming that a sensor reports a value '1' if one or more targets are detected in its sensing range and '0' otherwise; (2) topological integration based approaches [19], [20] aim to obtain the *expected* target count in sensor networks based on the topological integration theory; (3) clustering based approaches [7], [8], [21] intend to identify multiple non-overlapping clusters, each of which containing one or more targets; and (4) statistics based approaches [9], [22] rely on probability theory to estimate the number of targets. There also exists an effort that considers target counting in camera sensor networks [23]. Note that the binary sensing model and the topological integration model report the bounds and expectation values, respectively, while the performance of clustering based algorithms heavily relies on the integration and partition of the total target energy in the overlapping influence area, which results in coarse counting. Additionally, statistics based approaches usually require a large number of sensor readings to obtain reasonably good results.

The most relevant research to this work is reported in [7] and [8], as they all require the computation of the energy volume, and then divide it by the unit target energy

volume, in order to derive the estimated number of targets. The differences are that our approach does not cluster the sensor readings and employs the Monte Carlo simulation to estimate the number of targets.

The EBAM algorithm proposed in [7] first partitions all the sensors into non-overlapping clusters based on the locations and readings of the sensors and then estimates the number of targets in each cluster. An estimated energy landscape is built for each cluster and the total energy volume is computed as the sum of the energy volume of each Voronoi cell in the cluster. By dividing the total energy volume by the unit target energy volume, the number of targets per cluster can be estimated. As the energy volume per Voronoi cell is completely determined by the sensor reading, which is very sensitive to the distance between the sensor and the targets, a ceiling function is introduced to upper-bound the contribution of an individual sensor reading. However, this ceiling function may cause serious undercounting when the targets are dense. Therefore this approach is only suitable for the case when the targets are sparsely located.

On the other hand, [8] takes a polynomial regression model to build the estimated energy landscape for each cluster. Though the estimated energy landscape from the regression is smoother and better, it may result in energy leakage. Therefore the unit target energy volume is multiplied with a fixed justification factor (0.7 in [8]). However, the performance of this algorithm is very sensitive to the justification factor, which is hard to determine in general sensor network settings.

The Monte Carlo method [12], [13] has been widely used in solving problems from difficult calculus calculation, probability calculation, parameter estimation, etc. It provides a useful tool to model the phenomena with a significant input uncertainty and to study the systems with a large number of coupled degrees of freedom. In this paper, we employ the Monte Carlo simulation to build a target energy landscape in a virtual sensor field whose shape is close enough to that of the original sensor field. The estimated energy volume from the virtual energy landscape is used to compute an estimated number of targets in the original sensor field. Our approach does not require clustering, and no justification factor or ceiling function is utilized to calibrate the estimated target count. Nevertheless, it achieves superior performance for both sparse and dense target counting. By incorporating the sensor readings from multiple epochs, the counting accuracy can be further improved. In fact, we argue that our Monte Carlo based target counting algorithms approximately converge to the real count. In wireless sensor networks, the Monte Carlo method is employed to tackle the problem of target tracking and target localization in [24].

Our work can also be regarded as a type of statistics based approaches but it does not require a large number of sensors to count an even larger number of targets. Compressive

sensing (CS) based approach [22] relies on the CS theory to estimate the number of targets in sensor networks. This approach is suitable for sparse targets only. By deriving a probability mass function (pmf) of the total target counts and utilizing partition and compensation methods, [9] presents a work that deals with double-counting in sensing for wireless sensor networks. This work assumes that each sensor captures the number of targets under its sensing coverage while our work requires the direct measurements of the target energies.

III. BACKGROUND INTRODUCTION

In this section, we introduce our network model and the relevant background information.

Let $S = \{s_1, s_2, \cdots, s_{N_s}\}$ be the set of N_s sensors deployed in a two-dimensional (2D) monitored area R. Assume that the physical position of each sensor is known to us, which can be obtained by techniques such as those proposed in [25]–[27]. Denote by $T_g = \{t_1, t_2, \cdots, t_{N_t}\}$ the set of N_t mobile targets whose distribution in R is arbitrary at any instant of time. We also use t_i to represent the location of the target t_i when there is no ambiguity from context. The emission of the target signal amplitude follows a decay model captured by a function $f_{\mathcal{E}}(d)$, where d is the Euclidian distance to the target. For example, $f_{\mathcal{E}}(d) = \frac{1}{(1+d)^\alpha}$, with α being the decay factor in the range of $[2.0, 5.0]$, is a popular target signal amplitude decay model that has been widely adopted [7], [8]. Following the mainstream research [7], [8], we assume that the signal amplitude at any point p in R is the superposition of the signal amplitudes of all targets at that position:

$$\mathcal{E}(p) = \sum_{j=1}^{N_t} f_{\mathcal{E}}(d(p, t_j)). \quad (1)$$

The reading of a sensor s_i, denoted by $\mathcal{E}(s_i)$, equals $\mathcal{E}(p)$, where p is the position at which the sensor is placed. The *energy landscape (signal amplitude profile)* over the whole sensor field R, denoted by $\mathcal{E}(S, T_g)$, is given by:

$$\mathcal{E}(S, T_g)(p) = \mathcal{E}(p), \text{for } p \in R. \quad (2)$$

Note that sensors are usually sparsely deployed and each sensor provides only a local observation of the phenomenon in the monitored area. Therefore it is impossible to obtain a precise energy landscape based on the discrete sensor readings. In the following, we propose to employ Voronoi Cells to build an estimated energy landscape.

Given any sensor $s_i \in S$, its Voronoi Cell $V_c(s_i)$ consists of all the points in the sensor field that are not farther from s_i than to any other sensor, i.e. a point $p \in V_c(s_i)$ if and only if $d(p, s_i) \leq d(p, s_j)$ for any s_j with $s_j \neq s_i$. The estimated energy landscape over the sensor field, denoted by $\hat{\mathcal{E}}(S, T_g)$, is then given by

$$\hat{\mathcal{E}}(S, T_g)(p) = \mathcal{E}(s_i) \text{ for } p \in V_c(s_i). \quad (3)$$

The Monte Carlo simulation is a collection of different methods that perform experiments using random numbers and the probability theory to get approximate solutions to a specific problem. In this study, the Monte Carlo simulation is used to estimate the number of targets that could generate $\hat{\mathcal{E}}(S, T_g)$. We start from a virtual area R' that is exactly the same as R. R' also contains a set of N_s sensors, denoted by $S' = \{s'_1, s'_2, \cdots, s'_{N_s}\}$, with the position of s'_i being the same as that of s_i in R. Therefore we have $V_c(s'_i) = V_c(s_i)$ for $i = 1, 2, \cdots, N_s$. Initially $\mathcal{E}(s'_i) = 0$ for $\forall s'_i \in S'$. Virtual targets, denoted by $T'_g = \{t'_1, t'_2, \cdots, t'_{N'_t}\}$, which have exactly the same target signal amplitude decay property as those of the real targets, are added to R' one by one following a particular distribution until the "shape" of $\hat{\mathcal{E}}(S', T'_g)$, the estimated energy landscape of R', is close enough to that of R. Whenever a new virtual target is placed in R', all $\mathcal{E}(s'_i)$ values are updated based on the target energy decay function.

In the following two sections we detail our approach regarding how to insert virtual targets to R' and how to estimate N_t from $\hat{\mathcal{E}}(S, T_g)$ and $\hat{\mathcal{E}}(S', T'_g)$, where $\hat{\mathcal{E}}(S', T'_g)$ is generated by N'_t virtual targets. We consider single-epoch and multi-epoch sensor readings, respectively.

IV. MONTE CARLO TARGET COUNTING BASED ON SINGLE-EPOCH SENSOR READINGS

In this section, we present our method to employ the Monte Carlo simulation for estimating the total number of targets based on the sensor readings collected within a single-epoch. We assume that all target positions are independent of one another and follow the same distribution.

A. Building the Virtual Energy Landscape

We view a target position in the original sensor field as a random vector (X, Y), whose probability density function is $f(x, y)$. This function $f(x, y)$ is usually unknown. However, it can be estimated by $\hat{f}_{\mathcal{E}}(x, y)$, which equals $\hat{\mathcal{E}}(S, T_g)$ divided by the integral of $\hat{\mathcal{E}}(S, T_g)$ over R. Note that here we use $\hat{f}_{\mathcal{E}}(x, y)$ as an estimate of the distribution of the true target position within one epoch by treating the signal decay function (after scaling) as the kernel function and following the usual kernel smooth process. This known function $\hat{f}_{\mathcal{E}}(x, y)$ is used to generate the virtual targets and consequently establish the virtual energy landscape. The following is the detail.

We first compute the series of ratios $\frac{\mathcal{E}(s_1)}{\mathcal{E}_{max}}, \frac{\mathcal{E}(s_2)}{\mathcal{E}_{max}}, \cdots, \frac{\mathcal{E}(s_{N_s})}{\mathcal{E}_{max}}$, with $\mathcal{E}_{max}$ being the largest sensor reading. These ratios determine where a virtual target should be placed. Next, we repeat the following steps until N'_t (a predetermined integer) virtual targets are generated:

1) Generate a 2D random vector (x', y') according to the uniform distribution over R'. Without loss of generality, assume that (x', y') is located in $V_c(s'_i)$.

2) Generate $z_i \in (0,1)$ according to the uniform distribution over the unit interval.

3) A virtual target is placed at (x', y') in R' if and only if $z_i <= \frac{\mathcal{E}(s_i)}{\mathcal{E}_{max}}$.

These three steps constitute a Monte Carlo Accept-Reject sampling procedure, leading to N_t' virtual targets. Clearly, these N_t' virtual targets produce $\hat{\mathcal{E}}(S', T_g')$ that has a shape similar to that of $\hat{\mathcal{E}}(S, T_g)$ if N_t' is large enough.

Note that $\frac{\mathcal{E}(s_i)}{\mathcal{E}_{max}}$ represents the chance that a virtual target occurs at $V_c(s_i')$. Moreover, when the number of virtual targets becomes larger and larger, the distribution function $\hat{f}_\mathcal{E}'(x, y)$ (defined in a way similar to that of $\hat{f}_\mathcal{E}(x, y)$) of the location of a (random) virtual target approximates $\hat{f}_\mathcal{E}(x, y)$. Combing this with the previous discussion, we see that if both N_t' and N_t are large, $\hat{f}_\mathcal{E}'(x, y)$ is close to $f(x, y)$, the distribution function of the true target positions.

B. Estimating the Number of Targets

We note that the energy volume of $\hat{\mathcal{E}}(S, T_g)$, denoted by $\hat{V}(S, T_g)$, can be computed by summing up the product of each sensor reading and the area of the corresponding Voronoi Cell: $\hat{V}(S, T_g) = \sum_{i=1}^{N_s} \mathcal{E}(s_i) \cdot \text{Area}(V_c(s_i))$. This quantity $\hat{V}(S, T_g)$ represents an estimated energy volume of the original energy landscape based on T_g. Similarly, we define $\hat{V}(S', T_g')$ as the estimated energy volume of the virtual energy landscape based on T_g'.

Due to the property of superposition of target signal amplitudes, it is seen that $\hat{V}(S, T_g)$ can be decomposed into various contributions from N_t individual targets: $\hat{V}(S, T_g) = \sum_{j=1}^{N_t} \hat{V}(S, t_j)$. By the strong law of large numbers, $\frac{\hat{V}(S, T_g)}{N_t}$ converges almost surely to $E(\hat{V}(S, t))$ when $N_t \to \infty$. Here t is a random position of a true target and the operation E indicates expectation taken with respect to the probability density function $f(x, y)$. Therefore, if N_t is large, $\frac{\hat{V}(S, T_g)}{N_t}$ can be used to approximate $E(\hat{V}(S, t))$. Similarly, if N_t' is large, $\frac{\hat{V}(S', T_g')}{N_t'}$ can be used to approximate $E(\hat{V}(S', t'))$. Here t' is a random position of a virtue target and the operation E indicates expectation taken with respect to the density function $\hat{f}_\mathcal{E}'(x, y)$. From Section IV-A, when both N_t' and N_t are large, $\hat{f}_\mathcal{E}'(x, y)$ can be used to approximate $f(x, y)$, and hence $\frac{\hat{V}(S, T_g)}{N_t}$ is close to $\frac{\hat{V}(S', T_g')}{N_t'}$. Now seting $\frac{\hat{V}(S, T_g)}{N_t} = \frac{\hat{V}(S', T_g')}{N_t'}$ yields

$$\hat{N}_t = \frac{\hat{V}(S, T_g) \cdot N_t'}{\hat{V}(S', T_g')}. \quad (4)$$

Eq. (4) provides an estimate of N_t by using the size N_t' of the set of virtual targets and the estimated energy volumes of the original and virtual target fields. Algorithm 1 summarizes the procedure of our Monte Carlo target counting based on the sensor readings from one single-epoch.

Algorithm 1 MCTC-s: Target counting from single-epoch sensor data

1: Compute $\hat{\mathcal{E}}(S, T_g)$.
2: Run the Monte Carlo Accept-Reject sampling procedure until a stop criterion (e.g., $N_t' = 10,000$) is met.
3: Compute $\hat{N}_t$ based on Eq. (4).

V. MONTE CARLO MOBILE TARGET COUNTING BASED ON MULTI-EPOCH SENSOR READINGS

Algorithm 1 is designed to count the number of targets when only the sensor readings from a single-epoch are available. Sensor readings from a single-epoch constitute one snapshot of the behavior of the targets. Correspondingly, the estimate of the size of the set of targets can deviate significantly from the true size, especially when the size of the set of virtual targets is bounded due to certain reasons. To improve the estimate of the number of real targets, we consider multiple epochs that take into account the fact that targets are moving objects. Throughout the multi-epoch process we assume that the locations of targets are independently distributed according to a fixed distribution with a density function $f(X, Y)$.

A. The Algorithm MCTC-m

We assume that each sensor reports its reading to the central server once every epoch. As long as the number of targets stays unchanged, the newly available data can always be employed to help us better estimate the total number of targets. Note that in our consideration the position of each sensor stays still during its life span while the targets keep on moving. Therefore the sensor readings change from epoch to epoch. Let T_{max} be the maximum number of epochs to be considered. Then we can compute the estimated energy landscapes for all epochs: $\hat{\mathcal{E}}_1(S, T_g)$, $\hat{\mathcal{E}}_2(S, T_g)$, $\cdots$, $\hat{\mathcal{E}}_{T_{max}}(S, T_g)$. Correspondingly we can obtain the energy volume for each of the estimated energy landscapes, denoted by $\hat{V}_1(S, T_g)$, $\hat{V}_2(S, T_g)$, $\cdots$, $\hat{V}_{T_{max}}(S, T_g)$, respectively. Similarly, for each epoch i, we can build a virtual energy landscape $\hat{\mathcal{E}}_i(S', T_g')$, as described earlier, and compute its energy volume $\hat{V}_i(S', T_g')$. The question is then: how can we use the information collected at each epoch to estimate the size of the set of targets?

Our approach is to estimate the actual number of targets by using the total number of virtual targets generated in all epochs, the total estimated energy volume of the original sensor field, and the total estimated energy volume of the virtual sensor field. Let N_{it}' be the number of virtual targets at epoch i. Assume that the stop criteria of the Monte Carlo simulation is $N_{it}' = N_t'$, where N_t' is a constant. Then with a similar argument to that of Eq. (4), the number of the true

targets can be estimated as follows:

$$\hat{N}_t = \frac{\sum_{i=1}^{T_{max}} \hat{V}_i(S, T_g) \cdot N'_t}{\sum_{i=1}^{T_{max}} \hat{V}_i(S', T'_g)}. \tag{5}$$

In Eq. (5), $\sum_{i=1}^{T_{max}} \hat{V}_i(S, T_g)$ and $\sum_{i=1}^{T_{max}} \hat{V}_i(S', T'_g)$ are the sums of the estimated energy volumes for the original and the virtual target fields, respectively. Note that the formula in (5) is a direct extension of that in (4). Eq. (5) reduces to (4) when $T_{max} = 1$. The procedure of target counting based on the multi-epoch sensor readings is summarized by Algorithm 2.

Algorithm 2 MCTC-m: Target counting from multi-epoch sensor data

---

1: Run the Monte Carlo Accept-Reject procedure to compute the estimated energy volumes of both the original and the virtual sensor spaces for each epoch.
2: Compute $\hat{N}_t$ based on Eq. (5) .

In the following subsection, we analyze the performance of Algorithm 2 in terms of target counting accuracies.

B. Performance Analysis

Suppose in the ith epoch, the real targets are located at t_{ij} for $j = 1, 2, \cdots, N_t$, and the virtual targets reside at t'_{ij} for $j = 1, 2, \cdots, N'_t$. Decompose $\hat{V}_i(S, T_g)$ into $\hat{V}_i(S, t_{ij})$ and $\hat{V}_i(S, T'_g)$ into $\hat{V}_i(S, t'_{ij})$. Then we rewrite (5) as

$$\hat{N}_t = \frac{\frac{\sum_{i=1}^{T_{max}} \sum_{j=1}^{N_t} \hat{V}_i(S, t_{ij})}{T_{max} \cdot N_t}}{\frac{\sum_{i=1}^{T_{max}} \sum_{j=1}^{N'_t} \hat{V}_i(S', t'_{ij})}{T_{max} \cdot N'_t}} \cdot N_t \tag{6}$$

Assume that all the locations t_{ij} of the true targets are independent. Then $\{\hat{V}_i(S, t_{ij})\}$ and $\{\hat{V}_i(S', t'_{ij})\}$ are two random samples. As in the derivation of Eq. (4), we see that as $T_{max} \cdot N_t \to \infty$, $\frac{\sum_{i=1}^{T_{max}} \sum_{j=1}^{N_t} \hat{V}_i(S, t_{ij})}{T_{max} \cdot N_t}$ converges almost surely to the expectation $E(\hat{V}(S, t))$, where t is a random position of a true target and the expectation is taken with respect to the probability density function $f(x, y)$. We also see that as $T_{max} \cdot N'_t \to \infty$, $\frac{\sum_{i=1}^{T_{max}} \sum_{j=1}^{N'_t} \hat{V}_i(S', t'_{ij})}{T_{max} \cdot N_t}$ converges almost surely to the expectation $E(\hat{V}(S', t'))$, where t' is a random position of a virtue target and the expectation is taken with respect to the probability density function $\hat{f}'_{\mathcal{E}}(x, y)$. In addition, if both N'_t and N_t are large, $\hat{f}'_{\mathcal{E}}(x, y)$ can be used to approximate $f(x, y)$, so that $E(\hat{V}(S', t'))$ tends to approximate $E(\hat{V}(S, t))$, and thus the right hand side of (6) tends to approach N_t. The above says that when the size of the set of true targets and the size of the set of virtual targets are large, the estimated size of the set of true targets is close to the actual size.

We summarize the above into the following: When N_t, N'_t, $T_{max}N_t$, and $T_{max}N'_t$ are large, the estimate $\hat{N}_t$ from

Algorithm 2 is approximately unbiased for the true target count. Note that in general, the need of large values of N_t and N'_t comes from the requirement that the distribution of a random virtual target should approximate the distribution of a true random target. If the targets meet the piecewise uniformity, i.e., they are uniformly distributed within each Voronoi cell, the above conclusion holds true when only requiring T_{max} to be large.

We conclude this subsection by making the following remark: using the above argument, it is easy to see that when N_t and N'_t are large, the estimate $\hat{N}_t$ from Algorithm 1 is approximately unbiased for the true target count.

VI. PERFORMANCE EVALUATION

In this section, we report our simulation results to demonstrate the strength of our Monte Carlo simulation based target counting algorithms (Algorithms 1 and 2). The wireless sensor network under our study consists of N_s sensors and N_t targets on a $100m \times 100m$ area R, where N_s is drawn from $\{100, 225, 400\}$ and N_t is drawn from $\{10, 50, 100, 225, 400, 600\}$. For simplicity, we set the stop criterion of the Accept-Reject process to be $N'_t = 10,000$. We consider the following two scenarios:

1) single-epoch sensor readings
2) multi-epoch sensor readings

The first scenario tests the performance of Algorithm 1 while the second investigates the performance of Algorithm 2.

A. Scenario 1: Single-Epoch

In this simulation study, all the N_s sensors are randomly and uniformly deployed in R. The placement of the targets is either "Random", which indicates that all targets are randomly deployed in R, or "Hot-Spot", where targets are placed according to a 2D "Hot-Spot" distribution $\mathcal{N}(\mu_p, \sigma^2 I)$, with I being a 2×2 identity matrix, μ_p being fixed at the center of R and $\sigma = 20$. This "Hot-Spot" case simulates a target group scattered in a focused area (hot-spot). In fact, about 95% targets reside in a square that has the same center as R and has an area of $6400m^2$. Note that when generating targets according to $\mathcal{N}(\mu_p, \sigma^2 I)$ in our simulation, a target is simply dropped if it appears out of R. The target energy decay factor α is set to 2.0.

For comparison studies we consider the following two algorithms tailored from the EBAM algorithm in [7] and the regression algorithm in [8]:

- **VCTC.** VCTC stands for *Voronoi cell based Target Counting*, which is similar to MCTC-s except that no Monte Carlo simulation is performed in VCTC. To be specific, we estimate the number of targets by computing the energy volume for each Voronoi cell $V_c(s_i)$ and then dividing the total energy volumes of all Voronoi cells by $\mathcal{E}_t$, the unit energy volume contributed by a simple target. In other words, $\hat{N}_t = \frac{\sum_{i=1}^{N_s} \mathcal{E}(s_i) \cdot Area(V_c(s_i))}{\mathcal{E}_t}$.

- **Regression.** This algorithm computes a polynomial regression function from the sensor readings, which is used as the estimated energy landscape $\hat{\mathcal{E}}(S, T_g)$. Then the number of targets is estimated by dividing the total energy volume derived from $\hat{\mathcal{E}}(S, T_g)$ by $\mathcal{E}_t$. In other words, $\hat{N}_t = \frac{\int_R \hat{\mathcal{E}}(S, T_g)}{\mathcal{E}_t}$.

Note that VCTC is similar to EBAM [7] but it does not apply the ceiling function to upper-bound the contribution of each sensor reading. In fact, we have implemented EBAM in our simulation study and found that the ceiling function results in a significant undercounting when the number of targets is large. This could explain the reason why in the original study of EBAM [7], the authors consider the case of sparse targets where N_t is less than 20 but N_s is more than 100. The Regression algorithm is related to the one proposed in [8] but we do not utilize the scaling coefficient to calibrate the energy level of a single target. Our simulation study indicates that the scaling coefficient introduces a significant overcounting for dense target deployment.

We run the simulation 50 times and the averaged results are reported in Table I and II for the "Random" and "Hot-Spot" cases, respectively. The performance metrics to be investigated include the *Mean* and the *Average Absolute Error (AAE)* of the 50 runs, where $Mean = \frac{\sum_{i=1}^{50} \hat{N}_t^i}{50}$ and $AAE = \frac{\sum_{i=1}^{50} |\hat{N}_t^i - N_t|}{50}$, with $\hat{N}_t^i$ being the estimated target count at the ith run. Note that the Mean serves as an estimator of the true target count and the AAE can be regarded as a metric to measure the bias of the estimation.

From Table I and II we draw the following four important observations:

1) MCTC-s outperforms VCTC and Regression in terms of Mean and AAE for all settings, especially when the number of targets is large. The performances of VCTC and Regression are comparable for both target distribution patterns.

2) The Mean values indicate that VCTC and Regression always report a lower count for both target deployment patterns. Nevertheless, when targets are deployed in a "Hot-Spot" pattern, MCTC-s yields a slight over-count. Moreover, the undercounting problem of VCTC and Regression is much more serious than the under-counting or over-counting problem of MCTC-s.

3) The target deployment patterns slightly affect the performances of all the three algorithms. The "Hot-Spot" target pattern produces slightly better results.

4) Increasing the number of sensors does not obviously improve the counting accuracy for all the three algorithms under both target deployment patterns.

The first observation indicates that the Voronoi cells and the polynomial regression function used to compute the estimated $\hat{\mathcal{E}}(S, T)$ produce a similar estimation. On the other hand, the Monte Carlo simulation does play an important role for high-precision target counting especially when the number of targets is large. In MCTC-s, by introducing the virtual sensor field, the effect of sensing area boundary that may cause counting errors is largely removed when estimating the number of targets. From a theoretical point of view, MCTC-s intends to estimate the expected contribution of a single target to the observed sensor readings based on the unknown/hidden target distribution and the known sensor placement. In this process, the factors that might effect the estimation of energy volumes are indirectly counted. On the other hand, the estimated $\hat{\mathcal{E}}(S, T)$ introduces a larger error in both VCTC and Regression because the sensor readings drop sharply as the distance to a target increases. Therefore MCTC-s achieves a better result.

The second observation involves the issue of under-estimation and over-estimation. The overcounting problem of MCTC-s when targets are deployed in a "Hot-Spot" pattern results from the Normal distribution of the targets: some virtual Voronoi cells have low true target count but exhibit a high target energy because they are neighbors of the cells that have a much larger target count. The undercounting problem is a sign of energy leakage due to the sensor field boundary effect and the sharp decrease of the target energy w.r.t. distance. Notice that most existing target counting algorithms [7], [8], [18], [22] only consider the case where the targets are sparsely deployed. In such a case, the errors caused by the boundary effect and the sharp decrease of the target energy are not as severe as those in the dense target case. To overcome the undercounting or overcounting problem, many approaches introduce a parameter such as the ceiling function in [7] and the scaling coefficient in [8] to improve the counting accuracy. However, for different sensor layouts and different sensor field shapes, the setting of the parameter may be different and an optimal value is hard to obtain. Therefore as indicated in [7], [8], such a parameter can only help when the number of targets is small and the target deployment is sparse. Nevertheless, our MCTC-s algorithm does not need any regularity parameter but it yields more accurate results. Note that the overcounting and undercounting of MCTC-s is not severe because the Monte Carlo simulation indirectly considers the causal factors, demonstrating the strength of MCTC-s.

The third observation is in particular interesting. The "Hot-Spot" pattern produces slightly better results because the effect of the sensing area boundary is not significant as targets are deployed in the center area. The fourth observation indicates that it is unnecessary to deploy a large number of sensors for better counting accuracy because neighboring sensors observe similar phenomenon, which can be easily captured by energy volume estimation based on Voronoi cells or regression functions.

Table I
THE MEAN AND THE AAE WHEN THE TARGETS ARE RANDOMLY DISTRIBUTED IN R.

		$N_t = 10$		$N_t = 50$		$N_t = 100$		$N_t = 225$		$N_t = 400$		$N_t = 600$	
		Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE
$N_s = 100$	MCTC-s	9.7	1.6	49.8	3.3	96.4	5.3	218.5	8.1	390.9	13.0	588.9	14.0
	VCTC	8.7	2.0	43.8	6.2	84.5	15.5	191.8	33.2	342.3	57.7	513.4	86.6
	Regression	8.6	1.9	43.7	7.1	84.3	15.7	189.5	35.5	341.6	58.4	511.4	88.6
$N_s = 225$	MCTC-s	10.3	1.0	48.7	2.1	98.7	3.0	219.9	5.8	391.3	9.1	586.6	13.5
	VCTC	8.6	1.5	42.9	7.1	86.5	13.5	192.5	32.5	342.5	57.5	513.4	86.6
	Regression	8.6	1.5	42.4	7.6	85.4	14.6	191.9	33.1	342.5	57.5	511.9	88.1
$N_s = 400$	MCTC-s	9.8	0.5	51.1	1.6	97.6	2.8	220.0	5.1	391.4	8.7	586.2	13.9
	VCTC	8.5	1.5	43.8	8.1	86.4	13.6	192.8	32.2	342.7	57.3	512.8	87.2
	Regression	8.4	1.6	44.1	8.9	85.4	14.6	191.9	33.1	341.9	52.1	512.0	88

Table II
THE MEAN AND THE AAE WHEN THE TARGETS ARE DEPLOYED IN A "HOT-SPOT" PATTERN IN R.

		$N_t = 10$		$N_t = 50$		$N_t = 100$		$N_t = 225$		$N_t = 400$		$N_t = 600$	
		Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE
$N_s = 100$	MCTC-m	10.4	1.4	50.4	2.8	100.3	4.2	227.2	6.5	405.6	8.6.9	602.4	10.1
	VCTC	9.4	1.1	46.6	3.9	93.8	7.0	215.4	10.2	380.4	19.6	569.7	30.3
	Regression	9.2	1.2	47.8	3.0	95.9	5.1	214.5	10.9	379.8	21.1	564.8	35.4
$N_s = 225$	MCTC-m	10.0	0.6	50.2	1.9	101.0	2.5	225.6	3.0	402.4	5.8	602.9	5.7
	VCTC	9.3	0.9	47.3	3.0	94.6	5.4	213.0	12.0	378.4	21.6	567.7	32.3
	Regression	9.6	1.0	47.3	2.7	93.7	6.3	213.1	11.9	378.0	22.0	568.2	31.8
$N_s = 400$	MCTC-m	10.2	0.5	50.6	1.3	100.5	1.8	226.4	1.6	402.6	4.0	603.4	4.7
	VCTC	9.8	0.6	47.5	2.5	94.8	5.2	213.2	11.8	378.3	21.7	569.8	30.2
	Regression	9.5	0.8	47.3	2.7	94.4	5.6	212.4	12.6	379.0	21.0	568.4	31.6

B. Scenario 2: Multi-Epoch

In this subsection, we investigate the performance of MCTC-m (Algorithm 2). The number of epochs T_{max} is set to be 200. To eliminate the effects of sensor locations on the performance, we consider a grid structure in which a sensor is placed at the center of each grid. For a comparison purpose we also give the results when sensors are randomly and uniformly deployed in R. Each simulation scenario is repeated for 50 times and the averaged results are reported. The performance metrics are the Mean and the AAE defined in Section VI-A.

Our assumption for this study is that all target locations are independently and identically distributed according to a fixed distribution. This corresponds to the case with a high sensor sampling rate. We will investigate other cases in our future research. Similar to the single-epoch scenario, we consider Random and "Hot-Spot" target deployment patterns, which places targets randomly in R or using $\mathcal{N}(\mu_p, \sigma^2 I)$, with $\sigma = 20$ and μ_p being the center of R, respectively.

Note that investigating the sensor readings from multiple epochs to exploit their temporal correlation has been extensively studied for in-network information processing but not for target counting. To emphasize the strength of MCTC-m, we compare it with VCTC and Regression again. We run VCTC and Regression over the sensor readings of each epoch and output the running average of the estimated target counts at each epoch. Since both VCTC and Regression divide the total estimated energy volume by the single target

energy volume to get the number of targets, the multiple epoch version of VCTC or Regression actually estimates the number of targets based on the averaged estimated energy volume. The results are reported in Tables III to VI.

We notice that all the four observations obtained from the single-epoch scenario (tables I and II) hold true for the multi-epoch scenario. This is reasonable as in the multi-epoch scenario we assume that target distributions from epoch to epoch follow the rule of *i.i.d.* As expected, both the Mean and the AAE values are improved for all the three algorithms in the multi-epoch scenario. This phenomenon can be explained as follows. First, we achieve better results for VCTC and Regression because a much larger number of runs (200×50 for multi-epoch and 50 for single-epoch) are performed to obtain the averaged results. Second, as elaborated in Section V-B, MCTC-m converges to the true target count when T_{max} is large.

We also notice that when sensors are deterministically placed at the grid centers, all the three algorithms achieve better results compared to the case when sensors are deployed randomly and uniformly in R. This is because the initial sensor positions have some impact on the counting accuracy. Since sensor positions remain unchanged during the 200 simulated epochs, this impact is hardly removed.

To investigate the convergence of Algorithm 2, we plot the AAE values of $N_s = 100$ in Figs. 1(a) to 4(d). It is observed that MCTC-m quickly converges to a number that is close to the true target count (AAE approaches to a small value), which is consistent with our theoretical analysis, but VCTC and Regression demonstrate a significantly biased

Table III
THE MEAN AND THE AAE WHEN SENSORS ARE PLACED AT GRID CENTERS AND TARGETS ARE DEPLOYED RANDOMLY.

		$N_t = 10$		$N_t = 50$		$N_t = 100$		$N_t = 225$		$N_t = 400$		$N_t = 600$	
		Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE
$N_s = 100$	MCTC-m	10.0	0.0	49.1	0.9	98.5	1.7	221.4	3.8	393.6	6.5	590.3	9.7
	VCTC	8.7	1.1	43.2	7.0	86.3	13.6	194.3	30.6	345.7	54.5	518.4	81.6
	Regression	8.6	1.0	43.1	7.9	86.3	13.8	194.1	30.9	344.2	55.2	517.1	82.9
$N_s = 225$	MCTC-m	10.0	0.0	49.0	1.0	98.0	2.0	220.0	4.9	392.1	7.8	588.3	11.7
	VCTC	8.5	1.3	42.9	7.0	85.9	14.0	191.3	31.8	343.6	53.4	515.5	84.5
	Regression	8.6	1.1	42.9	7.0	85.8	14.0	193.1	31.9	343.2	56.9	514.7	85.2
$N_s = 400$	MCTC-m	10.0	0.0	49.0	1.0	98.0	2.0	220.3	4.8	391.6	8.4	587.3	12.7
	VCTC	8.6	1.0	42.8	7.0	85.6	14.1	192.7	32.1	342.5	57.5	513.8	86.3
	Regression	8.6	1.1	42.8	7.0	85.3	14.6	192.6	32.2	342.4	57.6	513.7	86.3

Table IV
THE MEAN AND THE AAE WHEN SENSORS ARE DEPLOYED RANDOMLY AND UNIFORMLY AND TARGETS ARE DEPLOYED RANDOMLY.

		$N_t = 10$		$N_t = 50$		$N_t = 100$		$N_t = 225$		$N_t = 400$		$N_t = 600$	
		Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE
$N_s = 100$	MCTC-m	9.9	0.0	49.0	1.1	97.9	2.1	221.2	4.0	392.4	8.0	587.7	12.3
	VCTC	8.6	1.3	42.7	7.2	86.1	13.8	193.5	31.5	344.6	55.4	517.8	82.2
	Regression	8.5	1.4	42.5	7.2	85.5	14.5	192.3	32.7	342.0	58.0	511.5	88.4
$N_s = 225$	MCTC-m	10.0	0.0	49.0	1.0	97.8	2.2	220.0	4.9	391.2	8.9	586.7	13.2
	VCTC	8.5	1.3	42.8	7.0	85.5	14.5	192.4	32.6	342.1	57.9	513.1	86.9
	Regression	8.5	1.3	42.7	7.1	85.3	14.7	191.8	33.2	340.9	59.1	511.6	88.3
$N_s = 400$	MCTC-m	10.0	0.0	48.9	1.0	97.9	2.0	219.9	5.1	390.9	9.3	586.5	13.5
	VCTC	8.5	1.3	42.7	7.0	85.4	14.6	192.2	32.8	341.8	58.1	512.4	87.6
	Regression	8.5	1.4	42.6	7.1	85.3	14.8	191.8	33.1	341.2	58.7	511.8	88.2

Table V
THE MEAN AND THE AAE WHEN SENSORS ARE PLACED AT GRID CENTERS AND TARGETS ARE DEPLOYED IN A "HOT-SPOT" PATTERN.

		$N_t = 10$		$N_t = 50$		$N_t = 100$		$N_t = 225$		$N_t = 400$		$N_t = 600$	
		Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE
$N_s = 100$	MCTC-m	10.0	0.0	50.3	0.3	100.7	0.7	226.5	1.5	402.7	0.7	604.1	4.1
	VCTC	10.0	0.0	47.5	2.6	95.0	5.0	213.5	11.5	379.5	20.5	569.2	30.8
	Regression	9.4	0.9	46.9	3.1	93.8	6.1	211.0	14.1	375.0	25.0	562.5	37.5
$N_s = 225$	MCTC-m	10.0	0.0	50.1	0.1	100.6	0.6	226.2	1.1	402.4	2.4	603.2	3.4
	VCTC	9.5	0.6	47.4	2.8	94.7	5.1	213.1	12.0	379.0	21.0	568.5	31.6
	Regression	9.4	0.8	47.1	3.0	94.3	5.9	212.1	12.9	377.0	23.1	565.6	34.3
$N_s = 400$	MCTC-m	10.0	0.0	50.1	0.0	100.5	0.6	226.1	1.2	402.0	2.0	603.1	3.1
	VCTC	9.5	0.8	47.3	3.0	94.7	5.1	213.1	12.1	378.8	21.2	568.2	31.9
	Regression	9.4	0.9	47.2	3.0	94.4	5.7	212.5	12.5	377.7	22.3	566.6	33.4

Table VI
THE MEAN AND THE AAE WHEN SENSORS ARE DEPLOYED RANDOMLY AND UNIFORMLY AND TARGETS ARE DEPLOYED IN A "HOT-SPOT" PATTERN.

		$N_t = 10$		$N_t = 50$		$N_t = 100$		$N_t = 225$		$N_t = 400$		$N_t = 600$	
		Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE	Mean	AAE
$N_s = 100$	MCTC-m	10.1	0.0	50.2	0.3	101.3	1.4	227.6	2.7	404.6	4.9	604.7	4.7
	VCTC	9.7	0.3	47.5	2.5	95.4	4.5	214.7	10.3	380.7	19.4	567.4	32.6
	Regression	9.4	0.6	47.6	2.6	94.0	5.9	211.5	14.0	379.5	20.9	562.5	37.5
$N_s = 225$	MCTC-m	10.0	0.0	50.1	0.1	100.7	0.7	226.3	1.3	402.5	2.5	603.6	3.7
	VCTC	9.5	0.5	47.4	2.8	94.8	5.1	213.2	11.8	379.0	21.0	568.6	31.4
	Regression	9.5	0.8	47.4	2.7	94.6	5.3	212.7	12.4	378.5	21.5	567.4	32.7
$N_s = 400$	MCTC-m	10.0	0.0	50.2	0.1	100.6	0.7	226.2	1.2	402.2	2.2	603.3	3.3
	VCTC	9.5	0.8	47.4	2.9	94.7	5.1	213.0	12.0	378.8	21.1	568.0	32.0
	Regression	9.5	0.6	47.3	2.8	94.6	5.3	212.9	12.0	378.5	21.6	567.3	32.7

result (AAE approaches to a much larger value). This is because via the Monte Carlo simulation MCTC-m captures the target distribution more accurately. Moreover, VCTC performs better when targets are deployed in the "Hot-Spot" pattern compared to Regression because Voronoi cells track the distribution of the targets more precisely compared to the polynomial regression.

VII. CONCLUSION AND FUTURE WORK

This paper proposes a Monte Carlo simulation based target counting algorithm to estimate the number of mobile targets in a sensor field. Our design relies on an Accept-Reject process to regulate the placement of virtual targets in
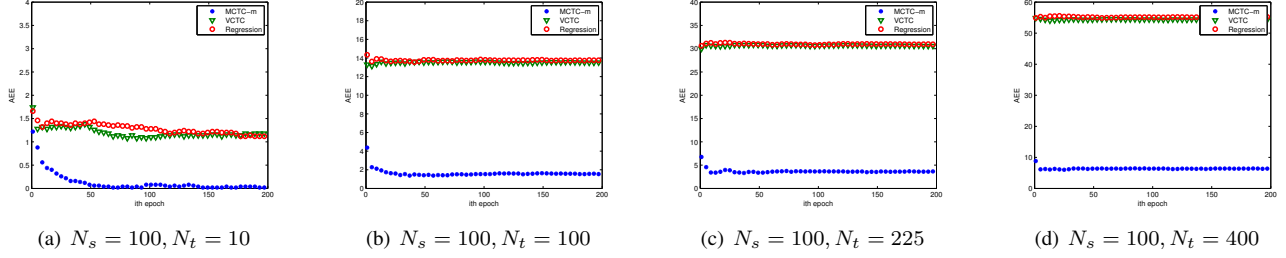
(a) $N_s = 100, N_t = 10$ (b) $N_s = 100, N_t = 100$ (c) $N_s = 100, N_t = 225$ (d) $N_s = 100, N_t = 400$

Figure 1. Sensors are placed at grid centers and targets are randomly deployed.

(a) $N_s = 100, N_t = 10$ (b) $N_s = 100, N_t = 100$ (c) $N_s = 100, N_t = 225$ (d) $N_s = 100, N_t = 400$
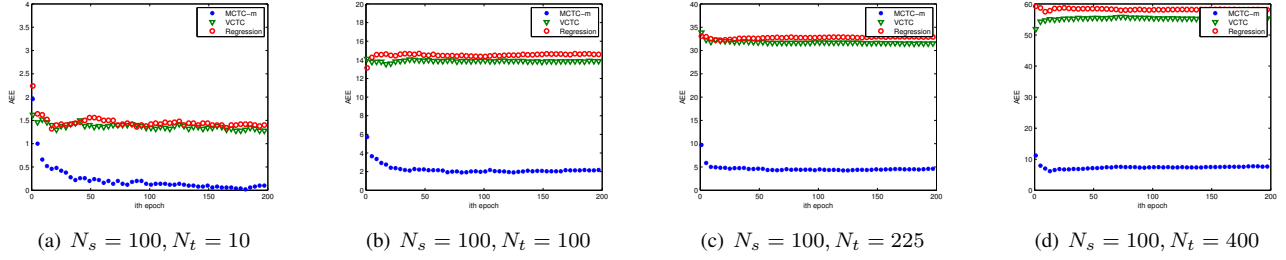
Figure 2. Sensors are placed randomly and uniformly, and targets are randomly deployed.

(a) $N_s = 100, N_t = 10$ (b) $N_s = 100, N_t = 100$ (c) $N_s = 100, N_t = 225$ (d) $N_s = 100, N_t = 400$
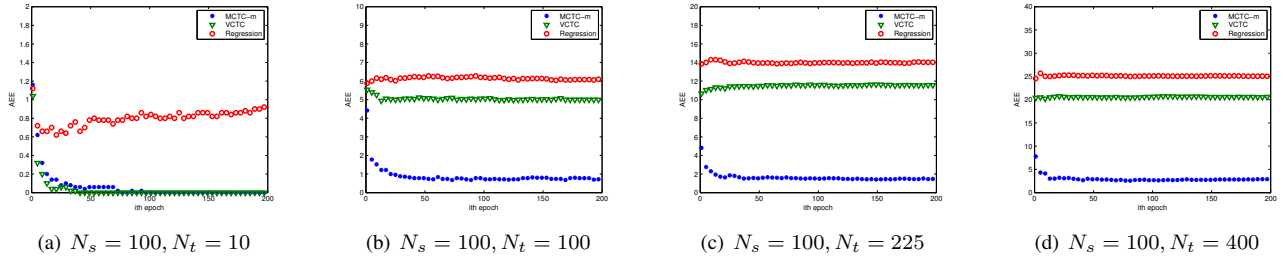
Figure 3. Sensors are placed at grid centers and targets are deployed in a "Hot-Spot" pattern.

(a) $N_s = 100, N_t = 10$ (b) $N_s = 100, N_t = 100$ (c) $N_s = 100, N_t = 225$ (d) $N_s = 100, N_t = 400$
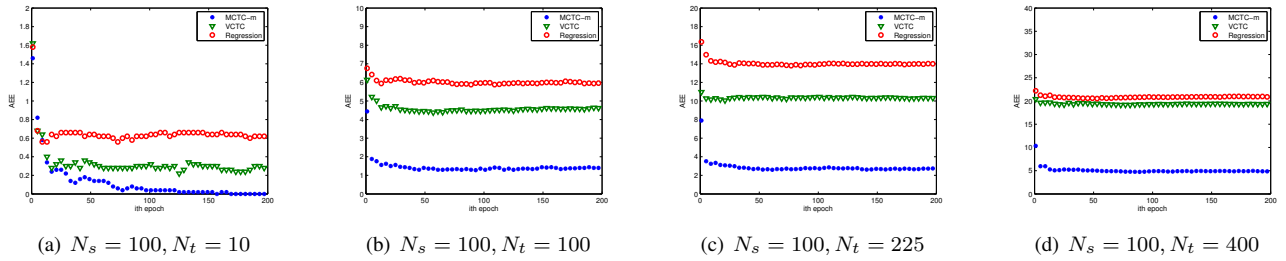
Figure 4. Sensors are placed uniformly and randomly, and targets are deployed in a "Hot-Spot" pattern.

a virtual field such that the shape of the virtual target energy landscape can mimic the one that is estimated from the real sensor readings. The role of the Monte Carlo simulation is to decrease the effects of the sensor field boundary and other hidden factors that could significantly affect the counting accuracy, as illustrated and justified in our simulation study. We conclude that the Monte Carlo simulation provides a powerful tool to effectively count the number of targets for both sparse and dense target deployments.

Our future research lies in the following three directions: i) we intend to design a better Accept-Reject procedure to more precisely capture the target deployment pattern; ii) we will provide both simulation study and theoretical performance analysis to investigate the case when target positions are correlated from epoch to epoch and the case when target distributions change from epoch to epoch; and

iii) we will employ regression to generalize our design to the case when target counts change from time to time.

REFERENCES

[1] M. Ding and X. Cheng, "Fault-tolerant target tracking in sensor networks," in *Proceedings of the 10th ACM international symposium on Mobile ad hoc networking and computing (Mobihoc'09)*, New Orleans, Louisiana, May 18-21 2009, pp. 125–134.

[2] Q. Liang, X. Cheng, and S. Samn, "New-catr: Network-enabled electronic warfare for collaborative automatic target recognition," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, no. 2, pp. 558–568, April 2010.

[3] M. Ding, D. Chen, A. Thaeler, and X. Cheng, "Fault-tolerant target detection in sensor networks," in *IEEE Wireless Communication and Networking Conference (WCNC)*, March 2005.

[4] Z. Wang, E. Bulut, and B. K. Szymanski, "Distributed energy-efficient target tracking with binary sensor networks," *ACM Transactions on Sensor Networks*, vol. 6, no. 4, July 2010.

[5] L. Liu, X. Zhang, and H. Ma, "Dynamic node collaboration for mobile target tracking in wireless camera sensor networks," in *IEEE INFOCOM*, 2009, pp. 1188–1196.

[6] S. Nikoletseas and P. Spirakis, "Probabilistic distributed algorithms for energy efficient routing and tracking in wireless sensor networks," *Algorithms*, vol. 2, no. 1, pp. 121 – 157, 2009.

[7] Q. Fang, F. Zhao, and L. Guibas, "Lightweight sensing and communication protocols for target enumeration and aggregation," in *In Proceedings of the 4th ACM International Symposium on Mobile ad hoc networking and computing*, 2003, pp. 165–176.

[8] Y. Guo, B. Hua, and L. Yue, "Energy-based target numeration in wireless sensor networks," in *FGCN '07: Proceedings of the Future Generation Communication and Networking*, 2007, pp. 380–385.

[9] S. Guo, T. He, M. Mokbel, J. Stankovic, and T. Abdelzaher, "On accurate and efficient statistical counting in sensor-based surveillance systems," *Pervasive and Mobile Computing*, vol. 6, no. 1, pp. 74 – 92, 2010.

[10] B. Gfeller, M. Mihalák, S. Suri, E. Vicari, and P. Widmayer, "Counting targets with mobile sensors in an unknown environment," in *Proceedings of the 3rd international conference on Algorithmic aspects of wireless sensor networks*, 2008, pp. 32–45.

[11] Y. Baryshnikov, E. Coffman, K. Kwak, and B. Moran, "Stochastic counting in sensor networks, or: Noise is good," in *Proceedings of the 4th IEEE international conference on Distributed Computing in Sensor Systems*, 2008, pp. 32–45.

[12] D. P. Rubinstein, R. Y.; Kroese, *Simulation and the Monte Carlo Method*, 2nd ed. John Wiley & Sons, 2007.

[13] N. Metropolis and S. Ulam, "The monte carlo method," *Journal of the American Statistical Association*, vol. 5544, no. 247, pp. 335–341, 1949.

[14] D. F. Watson, "Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes," *The Computer Journal*, vol. 24, no. 2, pp. 167–172, 1981.

[15] C. Robert and G. Casella, *Monte Carlo Statistical Methods*, 2nd ed. Springer-Verlag, 2004.

[16] W. Kim, K. Mechitov, J.-Y. Choi, and S. Ham, "On target tracking with binary proximity sensors," in *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, 2005, pp. 301–308.

[17] N. Shrivastava, R. Mudumbai, U. Madhow, and S. Suri, "Target tracking with binary proximity sensors," *ACM Transactions On Sensor Networks*, vol. 5, no. 4, pp. 1–33, 2009.

[18] J. Singh, U. Madhow, R. Kumar, S. Suri, and R. Cagley, "Tracking multiple targets using binary proximity sensors," in *Proceedings of the 6th international conference on Information processing in sensor networks*, 2007, pp. 529–538.

[19] Y. Baryshnikov and R. Ghrist, "Target enumeration via integration over planar sensor networks," in *Proceedings of Robotics Systems and Science*, 2008.

[20] ——, "Target enumeration via euler characteristic integrals," in *SIAM Journal on Applied Mathematics*, 2009.

[21] M. Ding, F. Liu, A. Thaeler, D. Chen, and X. Cheng, "Fault-tolerant target localization in sensor networks," *EURASIP Journal on Wireless Communnications and Networking*, vol. 2007, no. 1, pp. 19–27, 2007.

[22] B. Zhang, X. Cheng, N. Zhang, Y. Cui, Y. Li, and Q. Liang, "Sparse target counting and localization in sensor networks based on compressive sensing," in *IEEE INFOCOM*, 2010.

[23] M. Karakaya and H. Qi, "Target detection and counting using a progressive certainty map in distributed visual sensor networks," in *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, 2009, pp. 1 – 8.

[24] A. Baggio and K. Langendoen, "Monte carlo localization for mobile wireless sensor networks," *Ad Hoc Networks*, vol. 6, no. 5, pp. 718 – 733, 2008.

[25] X. Cheng, A. Thaeler, G. Xue, and D. Chen, "TPS: A time-based positioning scheme for outdoor wireless sensor networks," in *INFOCOM*, vol. 4, March 7-11 2004, pp. 2685–2696.

[26] W. Cheng, N. Zhang, M. Song, D. Chen, X. Lu, and Z. Lu, "Time-bounded essential localization for wireless sensor networks," in *Networking, Architecture and Storage (NAS), 2010 IEEE Fifth International Conference on*, 2010, pp. 3–12.

[27] Y. Wang, K. Li, and J. Wu, "Distance estimation by constructing the virtual ruler in anisotropic sensor networks," in *IEEE INFOCOM*, 2010, pp. 1–9.