A Model-based Approach for RFID Data Stream Cleansing

Zhou Zhao and Wilfred Ng
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
{zhaozhou, wilfred}@cse.ust.hk

ABSTRACT

In recent years, RFID technologies have been used in many applications, such as inventory checking and object tracking. However, raw RFID data are inherently unreliable due to physical device limitations and different kinds of environmental noise. Currently, existing work mainly focuses on RFID data cleansing in a static environment (e.g. inventory checking). It is therefore difficult to cleanse RFID data streams in a mobile environment (e.g. object tracking) using the existing solutions, which do not address the data missing issue effectively.

In this paper, we study how to cleanse RFID data streams for object tracking, which is a challenging problem, since a significant percentage of readings are routinely dropped. We propose a probabilistic model for object tracking in a mobile environment. We develop a Bayesian inference based approach for cleansing RFID data using the model. In order to sample data from the movement distribution, we devise a Gibbs sampler that cleans RFID data with high accuracy and efficiency. We validate the effectiveness and robustness of our solution through extensive simulations and demonstrate its performance by using two real RFID applications of human tracking and conveyor belt monitoring.

Categories and Subject Descriptors

H.2 [**Information Systems**]: Database Management

General Terms

Algorithms, Design, Experimentation

Keywords

Probabilistic Algorithms, Uncertainty, Data Cleaning

1. INTRODUCTION

RFID (Radio Frequency IDentification) technologies have been widely applied in many areas such as supply chains and warehouse management owing to its low cost and non-intrusive tracking techniques [6, 11]. However, raw RFID data are inherently unreliable due to physical device limitations and different kinds of environmental noise.

Most previous approaches for cleaning RFID data are rule-based inference algorithms [3, 10, 12, 18]. Although the methods arising from these approaches could be simple and fast, their accuracy is rather low. Currently, probabilistic model based approaches were proposed to cleanse RFID data and it can be shown that such approaches are better than those using rule-based algorithms [7]. Many model-based approaches [5, 17, 4, 7] also propose formal models for different RFID applications and cleanse data under the framework of *Expectation Maximization* or *Sampling*.

To clean RFID data collected from a mobile environment, we focus on the following three major issues:

- **Data Missing.** The read rate for RFID data in the real-world is often in the range of 60-70%, which means over one third of the data are missing [15, 9]. This poses a great challenge for mobile data cleansing because sometime, there is no observation of tracking objects.
- **Large Volume of Data.** The RFID data collected from mobile environment are always in quantity and arriving in high speed. Since these arriving data cannot be stored in the databases, we have to cleanse the data based only on current observation.
- **Real Time Inference.** Many RFID applications need the current locations of tracking objects in real time. For example, an elderly caring system monitors any abnormal behavior of an elderly and needs to inform a paramedic in real time.

There have already been some papers addressing RFID data cleansing by using probabilistic inference [5, 17, 4, 7]. However, none of them considers all the above issues.

In this paper, we study the problem of cleaning the RFID data streams in a mobile environment that causes large missing rate. The underlying idea of our approach for dealing with the data missing issue is to make use of the historic data of the tracking objects. Historic observations are able to provide some evidence to assist the location inference under the current timestamp. This paper presents a probabilistic model for RFID tracking objects in a mobile environment. Then a Bayesian inference based algorithm is proposed to sequentially clean the collected RFID data. Our model based approach is suitable for cleansing the RFID data stream in the mobile environment, our model takes advantage of the

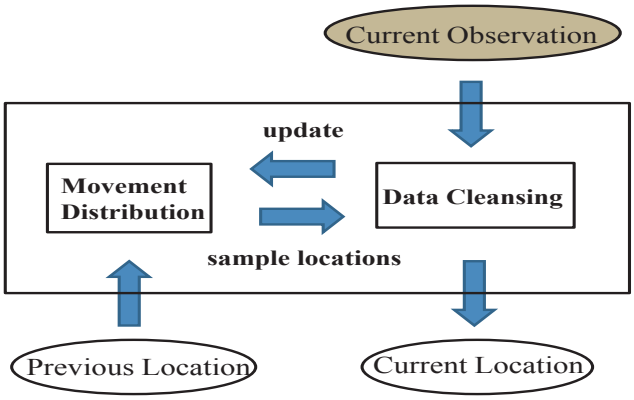


Figure 1: General RIFD Data Cleansing Process

spatiotemporal correlation of tracking objects. The general cleansing process for RFID data streams is depicted in Figure 1. Our model considers the movement of tagged objects in order to reduce the uncertainty of a missed reading.

Contributions. We mainly improve the utility of RFID data. We propose a probabilistic model to clean the RFID data collected in a mobile environment. We take advantage of the spatiotemporal correlation of tracking objects to tackle the missed reading problem. Specifically, we make the following contributions.

- We propose a probabilistic model for RIFD data stream cleansing in a mobile environment.
- We devise a Gibbs sampler to clean RFID data with high accuracy and efficiency.
- We employ extensive simulations to evaluate the robustness of our model and evaluate the effectiveness on two real RFID data, such as human tracking and conveyor belt monitoring.

This paper is organized as follows. Section 2 introduces the preliminary knowledge of RFID data and formulates the problem. Section 3 surveys the related work while Section 4 introduces a baseline algorithm to this problem. Section 5 then presents our model and states the sampling algorithm for cleansing data. Section 6 presents the experimental results and we conclude the paper in Section 7.

2. BACKGROUND

In this section, we introduce some background knowledge of RFID technologies and the notations used in our subsequent discussion of RFID data cleansing. Then, we formulate the problem of cleansing RFID data streams.

2.1 Preliminary Knowledge

RFID Technology. RFID is an electronic tagging and tracking technology designed to provide non-line-of-sight identification. The typical installment of RFID consists of three components: readers, antennae and tags. RFID readers communicate with tags using antennae. The antenna interrogates nearby tags by sending out an RF signal. Tags in the detection field respond to antenna by their unique identifier codes [16].

An acquisition of tags by an antenna in a static environment is composed of several reading iterations. Table 1 illustrates an example of an acquisition of 10 reading iterations by the antenna Ant_0. *Resp* denotes the number of responses received by the antenna during this acquisition. The reader rate is defined as the ratio of *Resp* to the total reading iterations sent by the antenna. For example, the read rate of tag "3008 33B2 DDD9 06C0 0000 0013" detected by Ant_0 is $\frac{7}{10}$ in Table 1.

TagID	Resp	AntID
3008 33B2 DDD9 06C0 0000 0013	7	Ant_0
3008 33B2 DDD9 06C0 0000 0012	6	Ant_0
3008 33B2 DDD9 06C0 0000 0005	2	Ant_0

Table 1: Tag Reading List for 10 Reading Iterations by Ant_0

Read Rate Distribution. We investigate the geographical read rate distribution of antennae by RFID readers and tags. The model of reader is Alien ALR-9900+[1] and the brand of RFID tag is Gen2 ALN-9640 "Squiggle" Inlay[2]. We put the RFID reader in the center of the room, then divide the area of the room into grids and put tags in the center of each grid. The RFID reader carries out the acquisitions for 15 minutes. The read rate distribution found is shown in Figure 2.

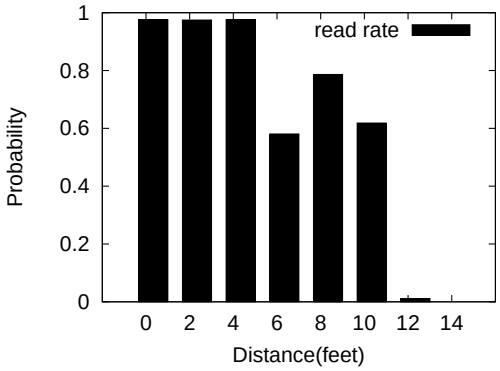


Figure 2: Read Rate Distribution

The read rate distribution plays an important role in RFID data cleansing such that the grids can be associated with detection probability. For example, Figure 2 shows that tags in the grids eight feet away from the antenna have a probability of 80% of being detected. The detection region is 12 feet in Figure 2.

We are also able to observe that the read rate may not always decay due to physical device limitations and different kinds of environmental noise, which is described by some specific curves, such as the *sigmoid* function [17]. In this paper, we employ grid-based discrete probability distribution to model the read rate deterioration. The grid size can be set according to the physical device and effect of environmental noise. We set the grid size to two feet for this work.

RFID Tracking. In a mobile environment, RFID antennae autonomously carry out reading iterations for moving

[1] `http://www.alientechnology.com/products/index.php`
[2] `http://www.alientechnology.com/tags/index.php`

TagID	Time	Resp	AntID
3008 33B2 DDD9 06C0 0000 0013	57:06.5	1	Ant_0
3008 33B2 DDD9 06C0 0000 0005	57:06.5	1	Ant_0
3008 33B2 DDD9 06C0 0000 0012	57:06.5	1	Ant_1
3008 33B2 DDD9 06C0 0000 0013	57:06.5	1	Ant_1

Table 2: Tag Reading List

tags during the tracking period and record the received responses by *Resp* at each timestamp.

We now give an example of a tag reading list in a mobile environment in Table 2. The *Resp* records the response of tags by 0 or 1. RFID antennae carry out one reading iteration at each 500ms. At 57:06.5, the tag "3008 33B2 DDD9 06C0 0000 0013" is detected by Ant_0 (first row) and Ant_1 (second row) simultaneously in Table 2. The tags not shown in Table 2 are considered missing at this timestamp.

2.2 Basic Concepts and Notations

DEFINITION 1. *The observed reading O is represented by a binary matrix which records the received response of tracking objects by antennae.*

The observed reading O is a binary matrix of two dimensions: tracking objects and antennae. Each entry in the matrix (i.e. o_{ik}) records the received response of object i by Ant_k which can be 0 or 1. Table 3 shows an example of the observed reading O. For instance, the response of object 1 (obj_1) is received by Ant_1(i.e. $o_{11} = 1$). We denote O^t to be the observed reading from the RFID data streams at the t-th timestamp.

	Ant_1	Ant_2	Ant_3	Ant_4
obj_1	1	0	0	0
obj_2	0	1	0	0
obj_3	1	0	0	0
...	...	...	...	...

Table 3: Observed Reading Matrix

As aforementioned, the tracking area is divided into grids. We represent the whole collection of grids as Z and denote each grid by z. The detection region of antenna can be said to be a set of grids with a positive read rate, denoted as R_k. The read rate distribution of Ant_k can be represented as $p(z|R_k)$ which is zero for $z \notin R_k$.

DEFINITION 2. *Given the observed reading O and a grid z, the posterior read rate $p(O|z)$ is the probability of the tracking objects as in the grid z.*

The relationship between observed readings and the posterior read rate can be categorized into four cases:

- If $o_{ik} = 0$ and $z \notin R_k$, then $p(o_{ik}|z) = 1$.
- If $o_{ik} = 0$ and $z \in R_k$, then $p(o_{ik}|z) = 1 - Pr(z|R_k)$.
- If $o_{ik} = 1$ and $z \notin R_k$, then $p(o_{ik}|z) = 0$.
- If $o_{ik} = 1$ and $z \in R_k$, then $p(o_{ik}|z) = Pr(z|R_k)$.

If obj_i is detected by Ant_k, then it must be at some grid z where $z \in R_k$. On the other hand, if obj_i is not detected by Ant_k, then it has a probability of $1-p(z|R_k)$ to be considered as a missed reading.

Table 4: Summary of Notations

Notation	Meaning
O	Observed Reading
L	Collection of Continuous Locations
Z	Collection of Discrete Grids
C	Grid Capacity
o_i	Observed Reading of obj_i
o_{ik}	Detection of obj_i by Ant_k
y_{it}	Location of obj_i at Time t
z_{it}	Grid of obj_i at Time t
$p(O\|z)$	Posterior Read Rate

2.3 Problem Definition

Using the notations given in Table 4, we define the problem of cleansing RIFD data streams in the mobile environment below.

DEFINITION 3. *Given a series of observed readings O^t where $t \in \{1, \ldots, T\}$, posterior read rate $p(O^t|z)$, we aim to find out which grid z the tracking objects are located in at each timestamp during the tracking period T.*

3. RELATED WORK

RFID data cleansing has attracted a lot of attention in the database community, which can be roughly classified into two categories: *rule-based inference* and *probabilistic inference*.

The rule-based inference algorithms for RFID data cleansing [3, 10, 12, 18, 8] were proposed in an early stage. These algorithms are directly applied to an RFID data stream or after the RFID data has been persisted. Examples of rules used are assigning the item to the first antenna which has identified it [12]. Another work from [18] assumes that the most recent data is correct and assigns the item to the last antenna that identified it. The item is assigned to the antenna with the most readings in [10]. The methods in the mentioned work are fast, but they also generate a lot of wrong predictions. Their accuracy and general performance do not outperform the probabilistic inference algorithm [7].

Recently, probabilistic inference algorithms were introduced as a new way of carrying out RFID data cleansing. The work in [13, 14] enabled declare query over RFID data streams of probabilistic events. The work [17] proposed to cleanse RFID data stream with reference objects such as shelves. The work [4] studied how to inference the containment relationship of tagged objects.

The most recent work [5] studied the RFID data cleansing problem and built a probabilistic model by taking capacity constraints of a location into consideration. A *Metropolis-Hasting* sampler based on the posterior read rate was proposed to infer the hidden variables in the model in order to get the locations of tagged objects. The experiments validate its performance in the static environment.

However, none of these data cleansing algorithms addresses the data missing problem arising from the RFID applications in a mobile environment such as object tracking, which involves a significant missing rate of the collected RFID data. Thus, we focus on studying how to cleanse such RFID data in a mobile environment. By taking capacity constraints [5] and data missing issues into consideration, we develop a new probabilistic model for object tracking.

4. BASELINE ALGORITHM

In this section, we first present a baseline algorithm (referred to as *BL*) for this problem. Our proposed algorithm considers the location inference at each timestamp independently and utilizes the posterior read rate to clean the RFID data streams. We only need to take the current RFID readings O^t and posterior read rate $p(O|z)$ into consideration. Then we infer the grids of tracking objects at time t. Formally, we denote a set of J location states $LS^0, LS^2, \ldots, LS^{J-1}$ to represent the possible grids in which the tracking objects are. The main procedure of *BL* is illustrated as the following three phases:

Initialization. The initial location state LS^0 of tracking objects are generated randomly. That is, for each tagged object, we select a grid to denote its location. Figure 3(a) is an example of the initial location state. Six grids are sampled for four objects in LS^0. The obj_1 in the first row is assigned to grid 1.

$$\begin{pmatrix}1&0&0&0&0&0\\1&0&0&0&0&0\\0&1&0&0&0&0\\0&0&1&0&0&0\end{pmatrix} \quad \begin{pmatrix}1&0&0&0&0&0\\0&1&0&0&0&0\\0&1&0&0&0&0\\0&0&0&1&0&0\end{pmatrix}$$

(a) LS^0 (b) LS^1

Figure 3: Location States

Update. For the current location state LS^j, we generate its neighboring state by applying a uniform distribution to it. For example, we have six grids of one dimension in Figure 3(a). The obj_4 is in grid 3 in LS^0 and the neighbors of grid 3 are grid 2 and grid 4. By using the uniform distribution, the possible grids for obj_4 in the next iteration are grids 2,3 and 4 with equal probability $\frac{1}{3}$. Figure 3(b) shows an example of LS^1 which is a neighboring location state of LS^0.

Selection. Suppose we denote the current location state as LS^j and its neighboring state as LS^{j+1}. We proceed to decide whether the neighboring state is accepted as the next state or rejected. By the known posterior read rate, the neighboring state LS^{j+1} is accepted with the probability $\frac{p(O^t|LS^{j+1})}{p(O^t|LS^j)}$. The probability $p(O^t|LS^{j+1})$ is the posterior read rate of state LS^{j+1} which can be factorized by

$$p(O^t|LS^{j+1}) = \prod_i p(o_i^t|LS_i^{j+1}) \tag{1}$$

where o_i^t is the RFID readings for obj_i at time t and LS_i^{j+1} is the sampled grid for obj_i. For example, in Figure 3(b), LS_3^1 is the third row(i.e. [0 1 0 0 0 0 0]). Finally, we get a set of location states for tracking objects in $\mathcal{S}$.

By considering the capacity of locations, the sampling process would become more effective and efficient. Suppose each grid has a capacity of C objects, then we are able to prune these location states directly by summing up their columns (i.e. $\sum_k LS_{ik}^{j+1} > C$).

Although it is feasible to clean RFID data streams, we may not be able to get high accuracy because of the missing data problem. In a mobile environment, there may be missed readings of tracking objects at some time due to physical device limitations and environmental noise. Under the missed reading scenario, the posterior read rate is not able to validate the quality of the sampled location state. Then the result of the baseline algorithm may be random and sharply deviates from the true location. We propose to make use of the spatial-temporal correlation of the tracking objects to tackle this problem.

5. ADAPTIVE CLEANSING MODEL

In this section, we present our *Adaptive Cleansing* model (referred to as *AC*). First, we introduce the general idea of our model. Second, we present the detail of our generative process. Third, we explain how to estimate the parameter.

5.1 General Idea

As aforementioned, the returned location states of *BL* are very uncertain for a significant missed reading of RFID data. Intuitively, to reduce any uncertainty, new constraints for cleansing RFID data are needed. For example, if we know the upper bound of the moving speed of some tracking object (i.e. radius of R grids), then we can sample its next possible locations with R grids.

One simple approach is to use a fixed radius R as the upper bound of the moving speed of tracking objects. The initial location state at time t is sampled from previous inferred locations by assuming a uniform distribution (i.e. $Uni(0, R)$). Thanks to the moving radius, the sampled location states becomes more effective such that the uncertainty is reduced. However, it is difficult to set the value of R. If R is too large, the returned location states are still very uncertain. On the other hand, if R is too small, then it fails to sample the correct location state and the accuracy of the algorithm becomes very low.

We propose a probabilistic model for the generation of the RFID data streams by considering its spatio-temporal correlation. The graphical representation of the adaptive cleansing model is given in Figure 4. We adopt the common motion model *Normal Distribution* [17] to generate the movement of tracking objects, denoted as θ. Formally, the location state at time t(i.e.L^t) is generated by the previous location and the motion model, denoted as $p_\theta(L^t|L^{t-1})$.

5.2 Generative Process

We now illustrate the generative process to produce the observed RFID readings $O^1, \ldots, O^T$ of tracking objects, given the posterior read rate. We denote a set of parameters $\varphi = \{\nu, \eta, \alpha, \beta\}$ as hyper parameters of our model. The procedure of generative process is outlined in Algorithm 1.

Algorithm 1 Generate Observed RFID Reading

Input: a set of tracking objects *obj*, a set of antennae and posterior read rate
Output: observed RFID reading O

1: Choose $\lambda \sim Ga(\alpha, \beta)$, $\mu \sim N(\nu, (\eta\lambda)^{-1})$
2: **for** $t = 1 \to T$ **do**
3: **for** For each object obj_i **do**
4: (a) Choose motion model $\theta_i \sim N(\mu_i, \lambda_i)$
5: (b) Choose location $l_{it} \sim p_{\theta_i}(l_{i(t-1)})$
6: (c) Locate the grid z covering location l_{it}
7: (d) Generate o_i^t by posterior read rate $p(O|z)$
8: **end for**
9: **end for**

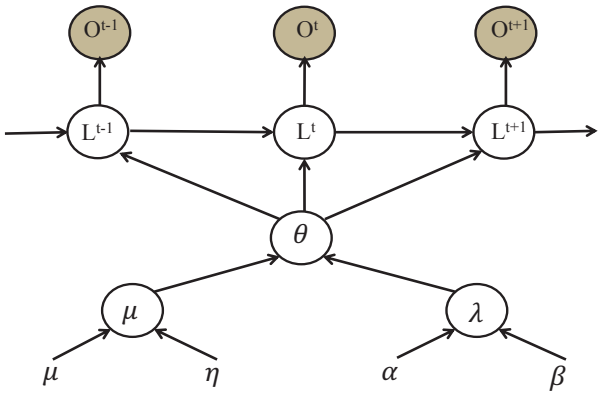


Figure 4: A Graphical Representation of The Adaptive Cleansing Model (AC)

5.2.1 Generating μ, λ

In order to generate motion model θ, we need to determine the average move parameter μ and the inverse variance λ.

We first sample λ from a Gamma distribution (i.e. $Ga(\lambda|\alpha,\beta)$). Given the sampled λ, we sample μ from a Normal distribution (i.e. $N(\mu|\nu,(\eta\lambda)^{-1})$). The joint distribution of μ, λ is given by

$$\begin{aligned} p(\mu,\lambda|\varphi) &= N(\mu|\nu,(\eta\lambda)^{-1})Ga(\lambda|\alpha,\beta) \\ &\propto \lambda^{\frac{1}{2}}\exp[-\frac{\eta_0\lambda}{2}(\mu-\mu)^2]\lambda^{\alpha-1}\exp(-\lambda\beta) \\ &= \frac{1}{Z}\lambda^{(\alpha-\frac{1}{2})}\exp\{-\frac{\lambda}{2}[\eta(\mu-\nu)^2+2\beta]\} \end{aligned} \quad (2)$$

where Z is the normalization factor of Equation 2.

5.2.2 Generating θ, L^t

Given the average movement μ_i and inverse variance λ_i of obj_i, we sample the location l_{it} from the motion model by $p_{\theta_i}(l_{it}|l_{i(t-1)})$. The location of all tracking objects is given by $L_t = [l_{it}]$.

5.2.3 Generating O^t

Given the current location for tracking object obj_i (i.e. l_{it}), the observed reading O_i^t of obj_i is generated from the posterior read rate by

$$\begin{aligned} p(O_i^t|l_{it}) &= \prod_k p(o_{ik}^t|l_{it}) \\ &= \prod_k p(o_{ik}^t|z) \end{aligned} \quad (3)$$

where k is an indication of the antennae and z is the grid covering the location l_{it}. $p(o_{ik}^t|z)$ is the posterior read rate of grid z. The current observed reading of all tracking objects is given by $O^t = [o_i^t]$.

Given the set of hyper-parameters $\varphi = \{\nu,\eta,\alpha,\beta\}$, we factorize the log likelihood of our model using the conditional independence assumption encoded in Figure 4.

$$\begin{aligned} \mathcal{L}(\varphi;O) &= \log p(O|\varphi) \\ &= \log\prod_{t=1}^{T}\prod_{obj_i} p(\theta_i|\varphi)p_{\theta_i}(l_{it}|l_{i(t-1)})p(o_i^t|l_{it}) \end{aligned} \quad (4)$$

5.3 Sequential Inference

The model in Figure 4 can be generalized as a kind of sequential probabilistic models [1]. *Particle Filter* [2] is a well known algorithm that solves sequential inference problems. However, it is difficult to apply this algorithm to our model inference problem because not only do we need to infer the hidden variables $L^1, L^2, \ldots, L^T$, but we also have to deduce the hyper model parameters $\{\nu,\eta,\alpha,\beta\}$.

We now devise a new sequential inference algorithm based on *Particle Filter* to solve the problem. Formally, we denote a set of samples (termed *particles* in the literature) at time t using $s_t^1, s_t^2, \ldots, s_t^J$, which is a hypothesis about the location of tracking objects. For the ease of the presentation, we use a particle as a location state in this section (i.e. a particle and a location state are interchangeable terms).

The set of initial particles $s_0^1, \ldots, s_0^J$ can be obtained from *BL*. The sequential procedure of our algorithm is:

- *Sampling.* For each particle s_{t-1}^j, we generate a new particle s_t^j from $p_\theta(s_t^j|s_{t-1}^j)$ where p_θ is the movement distribution of tracking objects.

- *Weighting.* We compute a particle weight as follows:

$$w_t^j = Bw_{t-1}^j \cdot \frac{p(O^t|s_t^j)}{p_\theta(s_t^j|s_{t-1}^j)} \quad (5)$$

  where B is a constant with respect to *j-th* particle, chosen so that $\sum_j w_t^j = 1$. $p(O^t|s_t^j)$ is the posterior read rate probability of particle s_t^j.

- *Re-sampling.* We sample the obtained particles to reproduce the highest weight ones. Each new particle is sampled from a set of old ones with replacement. The sampling probability of the particle is equal to its weight.

- *Re-estimating.* We compute the hyper model parameters by the obtained samples. The movement between two particles s_t^j and s_{t-1}^j can be denoted as $d^j = s_t^j - s_{t-1}^j$. Then m movements are sampled with probability proportional to $w_t^j w_{t-1}^j$, denoted as $D = \{d^1, \ldots, d^m\}$. The hyper parameter at time t (i.e.φ_t) is updated by φ_{t-1} and sampled movements D.

Now, we discuss how to estimate the hyper parameters sequentially. Then we discuss the location inference output of the tracking objects.

Recall the generating distribution for μ and λ by Equation 2, we set the generating distribution condition on φ_t to be equal to on D, φ_{t-1} as:

$$\begin{aligned} p(\mu,\lambda|\varphi_t) &= p(\mu,\lambda|D,\varphi_{t-1}) \\ &\propto Pr(\mu,\lambda|\varphi_{t-1})Pr(D|\mu,\lambda,\varphi_{t-1}) \\ &= Pr(\mu,\lambda|\varphi_{t-1})p(D|\mu,\lambda) \\ &\propto \lambda^{(\alpha_t-\frac{1}{2})}\exp\{-\frac{\lambda}{2}[\eta_t(\mu-\nu_t)^2+2\beta_t]\} \end{aligned} \quad (6)$$

where the updating schema for parameters φ_{t-1} by Equa-

tions 7, 8, 9 and 10.

$$\alpha_t = \alpha_{t-1} + \frac{m}{2} \quad (7)$$

$$\beta_t = \beta_{t-1} + \frac{1}{2}\sum_{i=1}^{m}(d_i - \overline{d})^2 + \frac{\eta_{t-1} m(\overline{d} - \nu_{t-1})^2}{2(\eta_{t-1} + m)} \quad (8)$$

$$\nu_t = \frac{\eta_{t-1}\nu_{t-1} + m\overline{d}}{\eta_{t-1} + m} \quad (9)$$

$$\eta_t = \eta_{t-1} + m \quad (10)$$

where $\overline{d}$ is the average movement of D. The derivation is according to the Bayesian rule and the details can be found in the Appendix. The time complexity of parameter re-estimation is $O(m)$.

The inference output of our model is a probability distribution of locations of tracking objects at any given time. Given a set of samples which are associated with their weights, the probability distribution of the locations is given by

$$p(z_t|O^t) = \sum_{j=1}^{J} w_t^j \mathbf{1}_{\{s_t^j \in z_t\}}. \quad (11)$$

where $\mathbf{1}_{a\in b}$ is an indicator function that is 1 if and only if the location of sample s_t^j is in the grid z_t. The grid z_t with the highest posterior probability is returned as the inference location of tracking object at time t.

5.4 Inference Algorithm

Now, we introduce our sequential inference algorithm for RFID data streams. Given particles s_{t-1}, model parameters φ_{t-1} at time $t-1$ and observed reading at time t, the algorithm re-estimates the model parameter and outputs a set of particles $\mathcal{S}_t$ at time t. Using particles $\mathcal{S}_t$, we are able to output objects' locations inference by Equation 11. The procedure of the algorithm is given by Algorithm 2 below:

Algorithm 2 Inference Algorithm(s_{t-1},φ_{t-1},O^t)

Input: $\mathcal{S}_{t-1}$: a set of particles at time $t-1$;φ_{t-1}: model parameters;O^t observed reading
Output: $\mathcal{S}_t$: a set of particles at time t;φ_t: current model parameters;

1: set $\mathcal{S} \leftarrow \emptyset$
2: **for** $j = 1 \rightarrow J$ **do**
3: **repeat**
4: particle $s_t^j \sim$ proposal distribution $p_\theta(s_t^j|s_{t-1}^j)$
5: **until** s_t^j subject to capacity constraint
6: weight $w_t^j = Cw_{t-1}^j \cdot \frac{p(O^t|s_t^j)}{p_\theta(s_t^j|s_{t-1}^j)}$
7: Add s_t^j to $\mathcal{S}$
8: **end for**
9: Re-sample J samples from $\mathcal{S}$ with replacement by their weights and add to $\mathcal{S}_t$
10: Compute movements D by $\mathcal{S}_t$ and $\mathcal{S}_{t-1}$
11: Re-estimate model parameter φ_t from D and φ_{t-1} by Equation 7, 8, 9 and 10.
12: **return** set $\mathcal{S}_t$ and parameter φ_t

The algorithm samples J qualified particles subject to capacity constraints from Line 3 to Line 5. Next, it associates sampled particles with weights at Line 6. It re-samples these particles to produce particles with the highest weights, denoted as $\mathcal{S}_t$, at Line 9. Then it re-estimates the model parameter φ_t at time t from D and φ_{t-1} by Equations 7, 8, 9 and 10 at Line 11. Finally, the algorithm produces a set of particles $\mathcal{S}_t$ and model parameters φ_t at time t.

Table 5: Default Values

Parameters	Values
z(Number of grids)	1500 (grids)
o(Number of objects)	50 (objects)
t(Number of timestamps)	100 (seconds)
c(capacity of grid)	4 (objects)
r(Number of grid by antenna)	5 (grids)
v(Moving speed of objects)	2 (feet per second)
δ(speed variance)	0.5
Missing rate	0.1

6. EXPERIMENT

We implement the proposed algorithms and study their efficiency and effectiveness using both real and synthetic data sets. The synthetic experimental evaluation is designed to investigate the robustness of our algorithm. All algorithms are implemented using Java. The experiments are performed in a Linux box with an 8-core Intel(R) Xeon(R) CPU X5450 3.00GHz and 16GB memory.

6.1 Synthetic Experiment

Synthetic Data Generation. We calibrate the read rate distribution of the synthetic data generator by posterior read rate collected from a static environment. We set the size of a grid to one square foot. The detection range of antenna is five grids. There is one grid overlapping in the detection range of two antennae. The size of tracking area is set to 1500 grids which is large enough for indoor environment. The movements of tracking objects are generated from *Normal Distribution* with *mean* two feet per second and *standard variance* 0.5. Other default values of the generator can be found in Table 5.

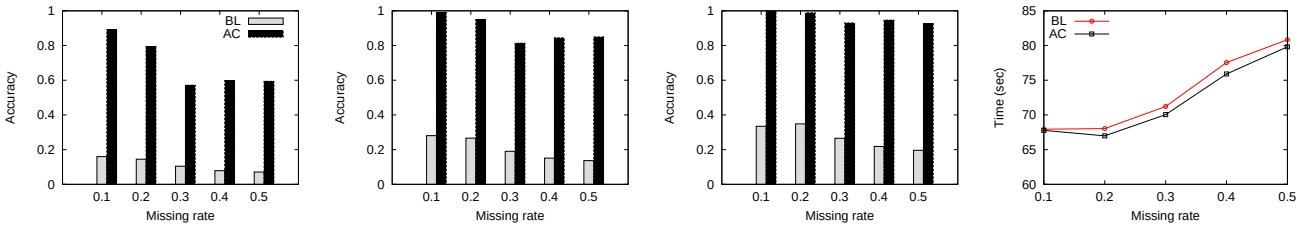
Measurement. We define $TopK$ accuracy to measure the effectiveness of the proposed algorithms. The inference output of the proposed algorithms is a probability distribution of locations. The locations with the top K highest probabilities are selected as the inference result. The $TopK$ accuracy is defined as the ratio of the number of correct cases in the inference result to the number of inference cases. The formula of $TopK$ accuracy is given by

$$TopK\,Accuracy = \frac{\#\text{ of correct cases}}{\#\text{ of objects} \times \#\text{ of timestamp}} \quad (12)$$

where the number of correct cases increases as K becomes large. In this experiment, we evaluate the effectiveness of the algorithms using Top1, Top2 and Top3 accuracy.

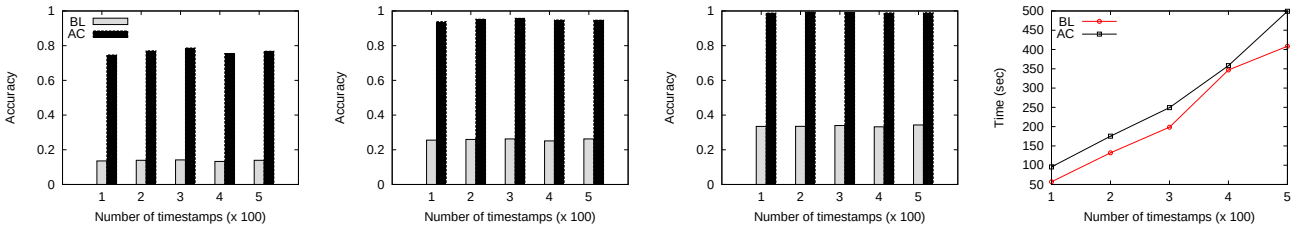
We compare the effectiveness and efficiency of BL with our proposed algorithm on five issues: (1) missing rate of RFID reading, (2) tracking time of the objects, (3) capacity constraint of the grid, (4) moving speed of the objects and (5) detection range of the antenna. The experimental result shows that our proposed algorithm is very robust.

Effect of Missing Rate. Figures 5(a), (b) and (c) illustrate the Top1,Top2 and Top3 accuracy of our algorithm on different missing rates (i.e. 0.1, 0.2, 0.3, 0.4 and 0.5),



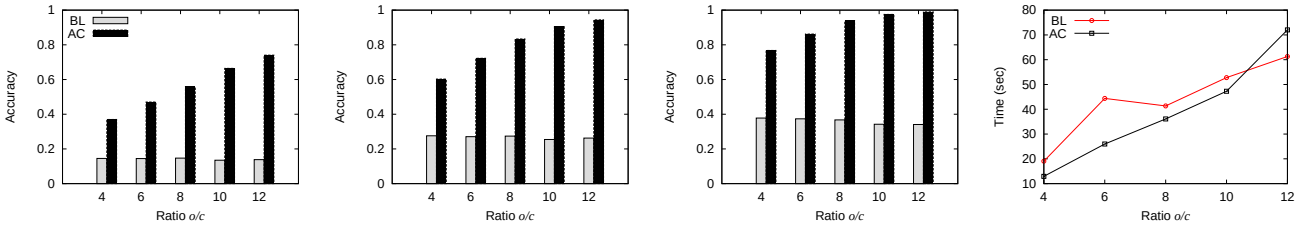
(a) Top1 (b) Top2 (c) Top3 (d) Running time

Figure 5: Effect of Missing Rate



(a) Top1 (b) Top2 (c) Top3 (d) Running time

Figure 6: Effect of Tracking Time



(a) Top1 (b) Top2 (c) Top3 (d) Running time

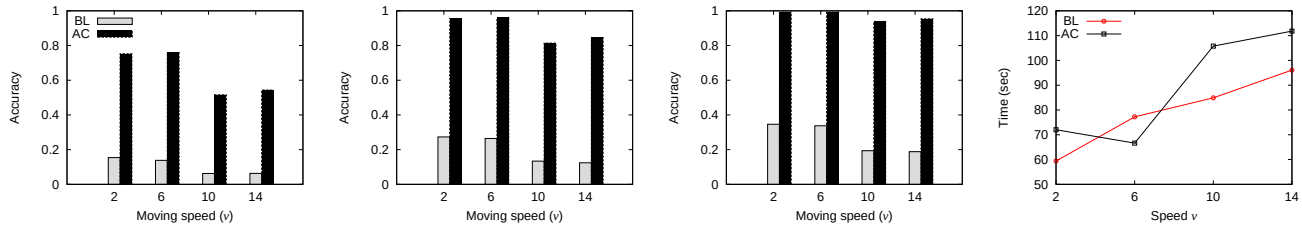
Figure 7: Effect of Objects by Capacity

respectively. For example, the missing rate 0.1 means that ten percentage of the RFID readings are missing during the tracking period.

As the missing rate increases, the inference accuracy of the algorithms deteriorates. However, the accuracy of the algorithm AC deteriorates slowly and outperforms the BL algorithm for all missing rates because our algorithm is able to capture the current movement of tracking objects in order to make the samples closer to the real moves. From Figure 5(a), we could observe that the algorithm AC has high $Top1$ accuracy when the missing rate is above 0.3, the common missing rate for RFID data. The $Top2$ accuracy shown in Figure 5(b) shows that AC has 80% inference correctness among these missing rates. The cost of the running time of our algorithm is slightly more efficient than BL, as shown in Figure 5(d). When the missing rate increases, it is difficult for BL to get qualified samples because the data uncertainty increases. However, our algorithm is able to utilize the estimated parameters to sample qualified particles in order to reduce the data uncertainty and improve the efficiency of the algorithm.

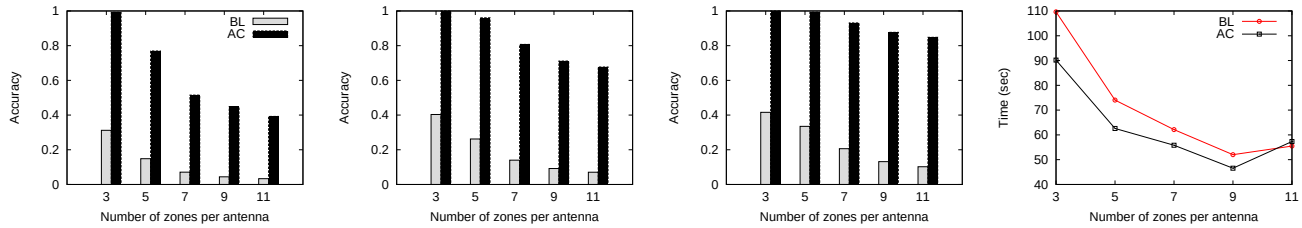
Effect of Tracking Time. We investigate the inference accuracy of the algorithm over different tracking time period to illustrate the robustness of our algorithm. Figures 6(a) to (c) demonstrate the Top1 to Top3 inference accuracy of our algorithm over different time periods in sec (i.e. 100, 200, 300, 400 and 500), respectively. We are able to observe that the inference accuracy of our algorithm is very stable, as shown in Figure 6. The Top2 accuracy of our algorithm reaches 90% for all the time periods in Figure 6(b). We could conclude that the inference accuracy of our algorithm does not decrease when we increase the tracking time period. So our algorithm is capable of cleaning RFID data streams in a satisfactory manner. The accumulated running time of the algorithms is given in Figure 6(d). The running time increases linearly. The time cost of location inference at one timestamp is less than one second which is very efficient.

Effect of Objects by Capacity. We study the effect of location capacity on our algorithm using a proposed ratio called *objects by capacity*. The *objects by capacity* is a ratio of the number of tracking objects by the location capacity. This constraint becomes stricter when we increase the number of tracking objects or reduce the location capacity. The strict constraint is able to prune many false positive candidate location states. For example, the location state can be considered to be invalid if it violates the constraint, no matter how high its probability is. Given the default grid capacity 4, we increase the number of tracking objects linearly. Figures 7(a) to (c) show the Top1 to Top3 accuracy result of our algorithm on different ratio of *objects by capacity* (i.e. 4,



(a) Top1 (b) Top2 (c) Top3 (d) Running time

Figure 8: Effect of Moving Speed



(a) Top1 (b) Top2 (c) Top3 (d) Running time

Figure 9: Effect of Grids per Antenna

6, 8, 10 and 12), respectively. The inference accuracy of our algorithm increases dramatically, as shown in Figure 7. The performance of our algorithm can be improved by *objects by capacity*. For example, the Top1 accuracy is below 40% when *objects by capacity* is set to 4 and it reaches 75% when the ratio set to 12 in Figure 7(a). The running time of the algorithms increases linearly, as shown in Figure 7(d).

Effect of Moving Speed. We investigate the inference accuracy of our algorithm by varying the speed of the tracking objects in (feet/s) (i.e.2, 6, 10 and 14). The inference accuracy of the algorithms decreases when we increase the moving speed, as shown in Figure 8. The sampling range of each move becomes larger as the moving speed increases. However, the inference accuracy of our algorithm only decreases slightly. The parameters of our model can be estimated by the sequential inference algorithm on the observed data. Then our model is able to capture the moving of tracking objects in order to make a better location inference.

Effect of Grids per Antenna. We study the effect of the density of antennae deployed by varying the number of grids managed by antenna (i.e. 3 ,5 ,7, 9 and 11 zones per antenna) in Figure 9. The inference accuracy decreases when we increase the grids managed by antenna (i.e. we decrease the density of antennae), as shown in Figures 9(a), (b) and (c). There are two reasons for this happening. First, the uncertainty of the observed readings increases when we assign more grids to the antenna. For example, suppose that the antenna only manages one grid. If the readings show that the tracking object is read by that antenna, we then know the object must be in that grid. If the antenna manages more grids, we only know the object is in one of its grids. Secondly, if the antenna manages more grids, the missing rate of the readings would increase because the read rate of the zones which are far away from the antenna is very low. Our algorithm is able to deal with this problem, since our algorithm is able to take the spatial-temporal correlation of the tracking objects which is very useful for reducing data uncertainty and coping with high missing rate.

The running time of the algorithm decreases first when the number of antennae is reduced. The algorithm only needs to process fewer antennae. Then the running time of the algorithm increases when we further increase the number of grids per antenna, as shown in Figure 9(d). This is because it is difficult to get qualified samples when the uncertainty of data raises.

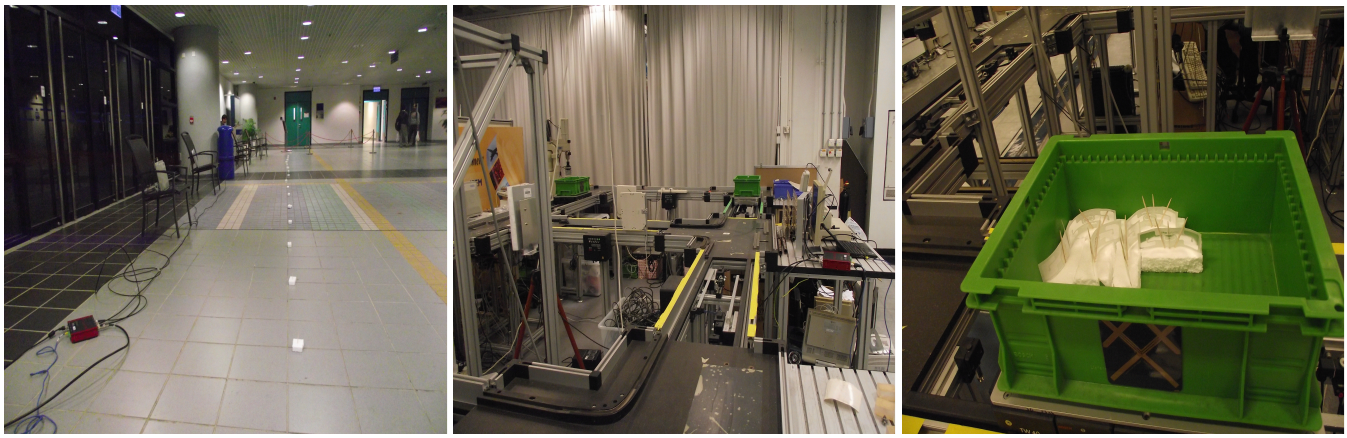
Through this experiment, we could conclude that our algorithm is very effective and more efficient than the *BL* algorithm on decaying with data uncertainty and missing issues. Our algorithm is based on a probabilistic model which takes the spatial-temporal consideration of tracking objects in order to reduce data uncertainty. Using the proposed sequential inference approach, our algorithm cleans the current RFID data in the streams and it only depends on the previous location states and estimated model parameters. Our algorithm is efficient enough to clean RFID data streams with high speed. The robustness of our algorithm is also evaluated through various issues.

6.2 Real Experiment

In this section, we evaluate the performance of the algorithms on different applications using two real datasets: human tracking and conveyer monitoring. We give a brief description of data collection and result analysis of human tracking and conveyor monitoring using our algorithm.

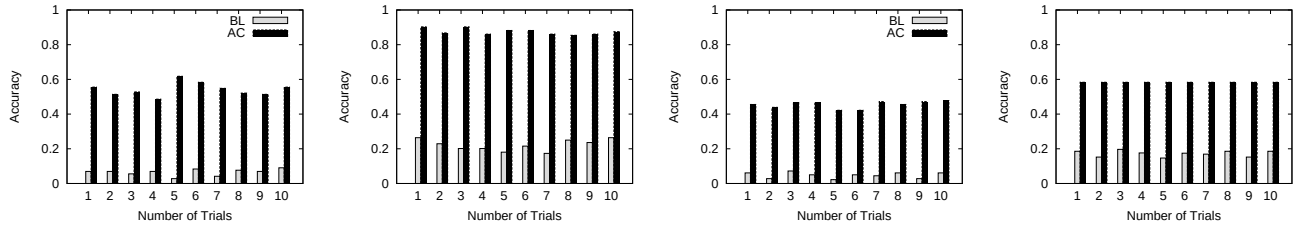
6.2.1 *Real Data: Human Tracking*

We evaluate the performance of our algorithm on the application of human tracking in this section. We use two *Alien* RFID readers and seven antennae in an indoor area. We divide the indoor area into 20 grids of two feet long. The capacity constraint of each grid is two in this experiment. We ask five undergraduate students to hold an RFID tag each and move inside this area. We ask the students to start at different grids inside this area. Then we record



(a) Human Tracking (b) Conveyor Monitoring (c) Transport Shuttle

Figure 10: Setting of Experiments



(a) Top1 (b) Top3 (c) Top1 (d) Top3

Figure 11: Experiments on Real Datasets

the RFID reading of these students for 40 seconds using autonomous mode of RFID readers. At the same time, we ask other students to record the actual grids of the tagged students passed during the tracking period as the ground truth of this experiment. Finally, we repeat this experiment 10 times. The deployment of antenna of this experiment can be found in Figure 10(a). Figures 11(a) and (b) illustrate the inference accuracy of our algorithm on 10 trials. We observe that our algorithm has more than 50% correctness in $Top1$ inference accuracy and 80% in $Top3$.

6.2.2 *Real Data: Conveyor Monitoring*

We validate the performance of our algorithm on the application of conveyor monitoring. We deploy four antennae on a Bocsh conveyor {`http://www.bosch.com/worldsite_startpage/en/default.aspx`}, as shown in Figure 10(b). We program the Bocsh conveyor to ask the transport shuttle to run on a predefined path at a constant speed. The whole path is 18 meters and we divide it into 18 grids of one meter long. The tagged objects are put in a transport shuttle, as shown in Figure 10(c). We consider the location of the transport shuttle as the location of the tracking objects. The ground truth of the movement of tagged objects is calculated based on the speed of transport shuttle and predefined path. We repeat this experiment 10 times. Each time, we start the transport shuttle at the same grid.

The inference accuracy of the algorithms can be found in Figures 11(c) and (d). We observe that the $Top1$ inference accuracy of our algorithm is more than 40% in Figure 11(c) which is much better than the *BL* algorithm. The inference accuracy reaches nearly 60% in Figure 11(d). The inference accuracy of this experiment is slightly lower than that of human tracking because the material of conveyors made of metal increases the false reading rate of the RFID antennae.

7. CONCLUSION

In this paper, we study the problem of cleaning RFID data streams in a mobile environment. To tackle the problem of significant missing rate, we propose a probabilistic model for RFID object tracking. Then a sequential inference algorithm is devised for inferring the hidden variables of our proposed probabilistic model. The sequential inference algorithm produces the inferred locations of tracking objects and incrementally re-estimates the model parameters in an efficient way. To evaluate the effectiveness and robustness of our proposed algorithm, we investigate its performance on various issues in the synthetic experiment. We also validate our algorithm on real data: human tracking and conveyor monitoring. The results of our algorithm on real data demonstrate the effectiveness of our algorithm to clean RFID data streams.

8. ACKNOWLEDGEMENTS

We thank the help from the UROP project student Gary Zhijun Zhang at HKUST for collecting the data used in our experiments. This work is partially supported by HKUST RFID Center under grant numbers ITP/022/02LP and SSRI08RGC, and RGC GRF under grant number HKUST 617610.

9. REFERENCES

[1] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *Signal Processing, IEEE Transactions on*, 50(2):174–188, 2002.

[2] C. Bishop and S. S. en ligne). *Pattern recognition and machine learning*, volume 4. springer New York, 2006.

[3] C. Bornhövd, T. Lin, S. Haller, and J. Schaper. Integrating automatic data acquisition with business processes experiences with sap's auto-id infrastructure. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 1182–1188. VLDB Endowment, 2004.

[4] Z. Cao, C. Sutton, Y. Diao, and P. Shenoy. Distributed inference and query processing for rfid tracking and monitoring. *Proceedings of the VLDB Endowment*, 4(5):326–337, 2011.

[5] H. Chen, W. Ku, H. Wang, and M. Sun. Leveraging spatio-temporal redundancy for rfid data cleansing. In *Proceedings of the 2010 international conference on Management of data*, pages 51–62. ACM, 2010.

[6] D. Delen, B. Hardgrave, and R. Sharda. Rfid for better supply-chain management through enhanced information visibility. *Production and Operations Management*, 16(5):613–624, 2007.

[7] L. Ferreira Chaves, E. Buchmann, and K. Böhm. Finding misplaced items in retail by clustering rfid data. In *Proceedings of the 13th International Conference on Extending Database Technology*, pages 501–512. ACM, 2010.

[8] H. Gonzalez, J. Han, and X. Shen. Cost-conscious cleaning of massive rfid data sets. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 1268–1272. IEEE, 2007.

[9] S. Jeffery, G. Alonso, M. Franklin, W. Hong, and J. Widom. A pipelined framework for online cleaning of sensor data streams. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 140–140. IEEE, 2006.

[10] S. Jeffery, M. Garofalakis, and M. Franklin. Adaptive cleaning for rfid data streams. In *Proceedings of the 32nd international conference on Very large data bases*, pages 163–174. VLDB Endowment, 2006.

[11] W. Ng. Developing rfid database models for analysing moving tags in supply chain management. *Conceptual Modeling–ER 2011*, pages 204–218, 2011.

[12] J. Rao, S. Doraiswamy, H. Thakkar, and L. Colby. A deferred cleansing method for rfid data analytics. In *Proceedings of the 32nd international conference on Very large data bases*, pages 175–186. VLDB Endowment, 2006.

[13] C. Ré, J. Letchner, M. Balazinksa, and D. Suciu. Event queries on correlated probabilistic streams. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 715–728. ACM, 2008.

[14] C. Ré, J. Letchner, M. Balazinksa, and D. Suciu. Event queries on correlated probabilistic streams. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 715–728. ACM, 2008.

[15] L. Sullivan. Rfid implementation challenges persist, all this time later. *Information Week*, 2005.

[16] F. Thiesse and F. Michahelles. An overview of epc technology. *Sensor Review*, 26(2):101–105, 2006.

[17] T. Tran, C. Sutton, R. Cocci, Y. Nie, Y. Diao, and P. Shenoy. Probabilistic inference over rfid streams in mobile environments. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*, pages 1096–1107. IEEE, 2009.

[18] F. Wang and P. Liu. Temporal management of rfid data. In *Proceedings of the 31st international conference on Very large data bases*, pages 1128–1139. VLDB Endowment, 2005.

10. APPENDIX

We give the derivation of model parameter re-estimation and show that the model parameter can be incrementally estimated.

Given a set of movements D and model parameter φ_{t-1}, we want to estimate model parameter φ_t at time t. Suppose that we have estimated model parameter φ_t, the the posterior distribution of μ and λ can be written as $Pr(\mu, \lambda|\varphi_t)$ which is equal to the posterior distribution given movements D and model parameter φ_{t-1}. The formula of this equation is given by

$$\begin{aligned}
Pr(\mu,\lambda|\varphi_t) &= Pr(\mu,\lambda|D,\varphi_{t-1}) \\
&\propto Pr(\mu,\lambda|\varphi_{t-1})Pr(D|\mu,\lambda,\varphi_{t-1}) \\
&= Pr(\mu,\lambda|\varphi_{t-1})Pr(D|\mu,\lambda) \\
&\propto \lambda^{(\alpha_t-\frac{1}{2})}\exp\{-\frac{\lambda}{2}[\eta_t(\mu-\nu_t)^2+2\beta_t]\} \\
&\propto \lambda^{(\alpha_{t-1}-\frac{1}{2})}\exp(-\frac{\lambda}{2}[\eta_{n-1}(\mu-\mu_{t-1})^2+2\beta_{t-1}]) \\
&\quad \cdot\lambda^{\frac{m}{2}}\exp(-\frac{\lambda}{2}\sum_{j=1}^{m}(d_j-\mu)^2) \\
&= \lambda^{(\alpha_{n-1}+\frac{m}{2}-\frac{1}{2})}\exp(-\frac{\lambda}{2}R)
\end{aligned}$$

where $R = \eta_{t-1}(\mu-\mu_{t-1})^2 + \sum_{j=1}^{m}(d_j-\mu)^2 + 2\beta_{t-1}$. It is clear that α_t is equal to $\alpha_{t-1}+\frac{m}{2}$. Then we study how to assign other model parameters. We first separate the second term in R(i.e. $\sum_{j=1}^{m}(d_j-\mu)^2$) into two components, as shown below:

$$\sum_{i=1}^{m}(d_j-\mu)^2 = m(\mu-\overline{d})^2 + \sum_{j=1}^{m}(d_j-\overline{d})^2$$

Then we rearrange R as:

$$\begin{aligned}
R &= \eta_{t-1}(\mu-\mu_{n-1})^2 + m(\mu-\overline{d})^2 + \sum_{j=1}^{m}(d_j-\overline{d})^2 + 2\beta_{t-1} \\
&= (\eta_{t-1}+m)(\mu-\mu_t)^2 + 2\beta_t
\end{aligned}$$

where we derive

$$\begin{aligned}
\beta_t &= \beta_{t-1} + \frac{1}{2}\sum_{j=1}^{m}(d_j-\overline{d})^2 + \frac{\eta_{t-1}m(\overline{d}-\mu_{t-1})^2}{2(\eta_{n-1}+m)} \\
\eta_t &= \eta_{t-1}+m \\
\mu_t &= \frac{\eta_{t-1}\mu_{t-1}+m\overline{d}}{\eta_{t-1}+m}.
\end{aligned}$$