2012 IEEE 13th International Conference on Mobile Data Management

Tracking the Dynamic Distribution of People in Indoor Space With Noisy Partitioning Sensors

Song Wang
Dept. of Computer Science
University of Vermont
swang1@cems.uvm.edu

X. Sean Wang
School of Computer Science
Fudan University, Shanghai, China
xywangCS@fudan.edu.cn

Yan Huang
Dept. of Computer Science
University of North Texas
huangyan@unt.edu

***Abstract*—The term "indoor" here refers generally to enclosed space partitioned into subspaces with connecting doors or gates. Examples include the inside of office buildings, amusement parks, and indoor shopping malls. In many applications, it is desirable to keep track of the distribution of people within the enclosed space. These applications range from smart house with automatically controlled air-conditioning and lighting, shopping assistance allocation, to business intelligence. Contact sensors are accurate but obstructive. Non-contact sensors such as automated visual recognition and RFID tags can be expensive in calibration or cost in order to obtain accurate readings. An interesting cost-optimization problem is to understand how to obtain relatively accurate dynamic distribution of people from inaccurate point sensors using correlations among the point sensor readings. The paper formalizes a framework that uses a flow model and a particle-filter learning algorithm for this optimization problem based on the hypotheses that (1) there is an underlying stochastic "flow model" of people moving within the space, and (2) with the help of this flow model, counting of people can be made more accurate by taking advantage of the continuous, albeit inaccurate, point sensor readings. The main challenge is that the performance of particle filters deteriorates rapidly with the number of doors in the indoor space. We propose a divide and conquer method that uses relatively more accurate sensors at a few strategically chosen locations to achieve overall good accuracy. Experimental results given herein show that the algorithm is effective.**

I. Introduction

Indoor space has become a new data management frontier recently [21], [9], [20], [10]. Keeping counts of people (people counting) in different indoor areas is a useful task in various applications including smart house with automatically controlled air-conditioning and lighting, shopping assistance allocation, and business intelligence. Contact counters such as rotation-bar can be accurate but obstructive. Non-contact sensors on the other hand may need extensive calibration, be sensitive to distance between continuous passing persons, or require near perfect detection zones. Although point sensors can be carefully installed and calibrated to count accurately, we believe an integrated approach where counter sensors "help" each other using a flow model would be a promising approach for people counting.

In this paper, we establish a stochastic dynamic framework that utilizes the correlation of sensor readings to achieve an overall accurate people counting system. We assume that doors are used to partition the space, and counting sensors are deployed at doors to report the number of people who have passed through the respective doors. Our research is based on the following three premises: (1) Counter sensors especially the non-contact ones are noisy and inaccurate. Even though they can be made accurate, it costs more and requires labor-intensive calibration. Our approach is in fact orthogonal to the research of improving individual counter accuracy; (2) There is an underlying stochastic "flow model" of people moving that we can utilize. This is a valid assumption because people can't move arbitrarily and are restricted by doors, i.e., which doors that a person can move through next is very much influenced by the door he passed through previously. This can be observed in a shopping mall area where the seemly "wandering" people go through stores with a stochastic pattern largely determined by her current location and demographics; (3) This flow model is represented by the probability distribution of the doors that the person is moving through next and can be obtained by various methods such as door distance and store type based distribution assignment, manual counting, and sampling using people wearing RFID tags.

This paper makes *three contributions* to the research in people counting. First, we take the first initiative to model the indoor people counting problem as a learning problem in the context of dynamic system. We propose a particle filtering based approach to estimate the distribution of people in indoor space. Our proposed model uses states, observations, and flow where "states" refers to the actual number of people walking through doors, observations are the sensor readings, and the flow model is for the stochastic transition between subsequent states. Estimating the states from observations (while given the transition model) is a not a new problem, however its use in people counting and our formulation are novel in some important respects. We place particular emphasis on maintaining a principled approach to the problem while simultaneously making the algorithm scalable. Due to the non-linear nature of the dynamic system we are modeling, we choose particle filters [14] over Kalman filter [12], [13] as the basic modeling framework. Our second contribution is our STAR model to partition the space into manageable subspaces to deal with the *curse of dimensionality problem* of the particle filter method. Specifically, we divide the space into subspaces and use more accurate (hence more costly) sensors for the doors that connect between the subspaces. The question then is how the subspace is obtained. Interestingly, after observing the general

978-0-7695-4713-8/12 $26.00 © 2012 IEEE
DOI 10.1109/MDM.2012.56

structure of indoor space, we stipulate that a large indoor space can generally be structured via a STAR model, i.e, several rooms form a star-connection (precise definition given later) to a "central room" (often hallways and staircases) and the central rooms are connected with each other. In our STAR model, we allow some exceptions, i.e., certain subspaces are connected directly without going through the central rooms. Finally we treat each star-connected component as a subspace which reduces the use of the more-costly sensors and gives good estimation results. Our third contribution is our extensive experiment to evaluate the performance of our particle filtering based method with the conclusion that our proposed method works well in most situations, under both small and large settings.

Organization: The rest of the paper is organized as follows: In Section II, we review the indoor data management and people counting methods. In Section III, we model the indoor space. We present our problem statement and assumptions in Section IV; review of general particle filters is presented in Section V. In Section VI, we detail the implementation of the particle filtering based algorithm; Section VII discusses the scalability of particle filtering and proposes the STAR model and in Section VIII, we present our experimental analysis of the algorithm. Finally we conclude with Section IX.

II. Related Work

Our work is generally related to work in three categories: indoor space data management, people counting, and dynamic system estimation. The last of the three is a classic field and the foundation of our proposed framework, we will introduce the basic concepts of particle filtering in details as we need. Below we will only summarize some related work of the two former categories.

A. Data Management in Indoor Space

Indoor has become a new data management frontier recently ([9], [10], [11], [20], [21]). Graph model based indoor tracking was discussed in [10]. In this model, users carry RFID tags while walking in the indoor space. Indoor space was represented using several graphs, such as base graph and deployment graph based on placement of RFID readers. Jensen et al [11] studied the trajectory indexing of moving objects in symbolic indoor space. Assuming trajectories of moving objects in indoor space can be collected by RFID readers, they proposed two R-tree based structures to index the trajectories. Range monitoring of moving objects and *k*NN queries were studied in [20] and [21], respectively. All these works used a graph model to represent the indoor space as we do in this paper, but they do not deal with people counting.

B. People Counting

For indoor space people counting, a lightweight people counting and localizing method using camera sensor nodes was proposed in [18]. By using motion and size criteria, they proposed to use a histogram to filter moving objects in indoor space within a specified size range with the deployment of sensors and cameras. People counting using multi-sensing application was proposed in [7]. They discussed the technique of using multiple sensors to detect the number of people passing and their moving directions through a *single door*. They can reach a counting accuracy with more than 95% in a 200cm wide door. Conard et al discussed a real-time people counter in [4]. By taking images in a reasonably restricted traffic flow such as in the entry of a building, it can count people with 95% accuracy in a very crowded traffic environment. While 95% is a good accuracy, we aim at using less expensive and less calibration intensive sensors to achieve similar results using readings from multi-doors. It is unclear how to extend their approaches to a multi-door context. A ram based neural network approach to count people was proposed in [17] for the purpose of predicting building usage patterns. They count people by taking images, while the neural network was used to process the images, i.e., recognize background scene. Our work differs from these works as we take these counters as given while aiming for reducing noise based on the dynamic movement of people through multiple doors in the indoor space. Our work can use these "spot" counters to achieve our goal.

Keeping accurate count of the number of people in outdoor space with the deployment of a wireless sensor network has been studied in [3]. Chow et al proposed a histogram based approach to answer aggregate location queries, i.e., how many people are located in a given region, by maintaining a histogram according to the reported counts from those "counting sensors". Several strategies are proposed to update the histogram. The feasibility of this histogram-based approach relies on the fact that in outdoor space, the cell towers can estimate how many people are in their covering region. Using counting sensors can simulate this functionality of the cell tower. It is not clear how to extend the histogram approach to indoor space. The reason is two-fold. Firstly, we generally do not have region counting sensors in indoor space. Secondly, indoor space usually cannot be modeled by a regular grid data structure (which is employed by the histogram based approach) as the size of a room and shape of rooms may be quite different. The advantage of indoor space is the more restricted movement of people that we intend to take advantage of (while [3] only uses the assumption that people generally don't move too fast).

III. Modeling the Indoor Space

Since our goal is to track the dynamic distribution of people in indoor space, it is critical to model the indoor space. In general, we model the indoor space as a graph $G = (V, E)$, where V represents the set of spaces, e.g., rooms, hallways and stairways; E are the connectors, e.g., doors between them. We track the distribution of people by placing sensors on doors.

For the purpose of presentation, we use the floor plan shown in Figure 1 as an example throughout this paper.

In general, a floor plan is comprised of rooms, doors, hallways and stairways. For simplicity reasons, we view hallways and stairways as ordinary rooms but having irregular shapes.

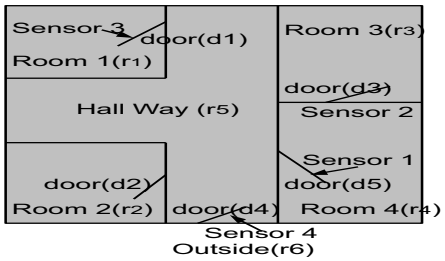


Fig. 1. An example of floor plan

We also view outside as a special room. Formally, a floor plan is defined as follows:

Definition (Floor Plan): A floor plan is a graph $G = (V, E)$, where

- V is the set of nodes representing rooms as well as hallways, stairways, and outside space;
- E is a set of edges connecting rooms;

In Figure 1, there are a total of six rooms r_1, r_2, r_3, r_4, r_5 (the hallway), r_6 (the outside space) and five doors. There are four door sensors s_1, s_2, s_3 and s_4 placed on d_1, d_3, d_4, and d_5. Sensors are assumed bidirectional, i.e., they can detect the number of people going either direction and report two respective counts. According to the graph model, we have $V = \{r_1, r_2, r_3, r_4, r_5, r_6\}$ and $E = \{d_1, d_2, d_3, d_4, d_5\}$.

In practice, it may not be practical, nor necessary, to place a sensor for each door based on our need of keeping track of where people are. The sensor placement induces the concept of "cells" that are unions of rooms that are not sensor partitioned. More specifically,

Definition (Indistinghuiability): Given two rooms r_i and r_j in V, if a person can move from r_i to r_j **without** being detected by any sensor then they are said to be *indistinguishable*.

Since doors are all bidirectional, indistinguishability is an equivalent relation over V, and each equivalence class consists of rooms within which people may move hitherto while avoiding being detected. Each equivalence class is called a "cell". For the purpose of counting people, it is only possible to estimate the people in a cell instead of in a room.

As an example, in Figure 1, r_2 and r_5 (the hallway) are an equivalence class since there is no sensor on door 2, and hence form a cell we call "c_0". Room r_1 forms its own cell (c_1), room r_3 is cell c_2, room r_4 is cell c_3, and finally room r_6 (the outside) forms c_4. Therefore, we have 6 rooms, but only 5 cells. The cells can be shown as a cell configuration graph, formally defined as follows:

Definition (Cell Configuration Graph (CCG)): Given a floor plan graph $G = (V, E)$, a cell configuration graph is a graph $G_c = (V_c, E_c)$, where

- V_c is the set of equivalence classes induced by the above indistinguishability relation from V,
- $E_c = \{e = (c_i, c_j) \in V_c \times V_c |$ there exist $r_i \in c_i$ and $r_j \in c_j$ such that $(r_i, r_j) \in E\}$.

Doors often allow bidirectional movement, i.e., people can go in either direction through a door. However, there are situations where doors are unidirectional, e.g., airport security entries. In order to accommodate these different scenarios, we assume all doors are unidirectional and a bidirectional door is modeled as two unidirectional doors. In Figure 1, we assume all doors are bidirectional and each is represented by two doors in either direction. Figure 2 gives the CCG induced from Figure 1.

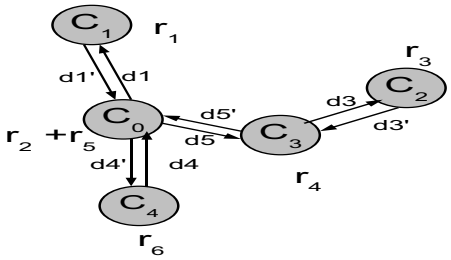


Fig. 2. An example of cell configuration graph induced from Figure 1

Based on the definition of CCG, we aim to track the distribution of people in indoor space among the cells. By definition of CCG, this distribution can be obtained by using the number of people that pass through all doors at a given time step.

IV. Assumptions and Problem Statement

Based on the aforementioned indoor model, we now present our formal assumptions and the problem statement.

Assumptions:

Based on the CCG, for each time k, we have an N-vector of underlying accurate readings, i.e., the actual number of people that pass the doors. (We assume that all doors are bidirectional in this work and hence each door in the floor plan corresponds to two doors in either direction with two corresponding sensors.) Formally, we name this N-vector as a "state", denoted as X_k. Specifically,

$$X_k = (x_k^1, x_k^2, \ldots x_k^N)^T$$

At each time k, since we have N partitioning sensor readings on those N unidirectional doors, we have another N-vector of sensor readings. In Figure 2, there are 8 unidirectional doors, so $N = 8$. Specifically,

$$Y_k = (y_k^1, y_k^2, \ldots y_k^N)^T$$

where each y_k^i is the sensor count for how many people passed a unidirectional door at time k. We denote Y_k as the "measurement" for time step k.

In practice, a sensor will generate Y_k. As usually found in the literature, we assume that there is a measurement function that can map a given state vector X_k to a corresponding measurement vector Y_k.

Measurement Function We assume the following measurement function:

$$y_k^i = x_k^i + m_i; \tag{1}$$

where m_i is the stochastic measurement noise for door d_i. Since we used low-cost noisy sensors, it is normal to assume that a Gaussian noise model exists inherent in those sensors.

From here on, unless otherwise explicitly explained, we will use $X_k[i]$ and $Y_k[i]$ to denote x_k^i and y_k^i, respectively. Besides,

we use $Y_{1:T}$ to represent those measurements from time 1 to time T. Meanwhile, we use $\mathcal{M}[i]$ to denote m_i, where $\mathcal{M}$ is the noise vector for all doors.

In the real world, people move around in the indoor space. However, people do not or cannot walk arbitrarily. For example, in a shopping mall, customers generally have different tendency to visit different stores based on, perhaps, the popularity of the stores and perhaps their age and gender. Therefore, the probability of passing some doors may be with higher probability than that of passing others. Motivated by this observation, we assume that when people who passed one door at k, the probability to pass other doors at $k+1$ is a multinomial distribution. We name this distribution for all doors $d_i (i = 1, ..., N)$ as a "flow model".

Flow Model Formally, we assume that the movement of people is decided by an underlying stochastic flow, it is a probability distribution represented in equation (2).

$$\Gamma = p(X_{k+1}|X_k), \tag{2}$$

where X_k and X_{k+1} are N-vectors at time k and $k+1$, respectively. $p(X_{k+1}|X_k)$ is a probability function that characterizes a given N-vector X_k evolving to another N-vector X_{k+1}. Generally, this system transition function is assumed to be known. This "flow model" may be obtained by tracking a sample of people wearing RFID tags or by manual counting. Since the preferences of user movement are seldom uniform, it make sense to assume that this flow model is a multinomial distribution for any given door that a person just walks by. Denoting the flow as Γ, it can be represented using the following probability matrix:

$$\Gamma = \begin{pmatrix} pr_{11} & pr_{12} \cdots & pr_{1N} \\ pr_{21} & pr_{22} \cdots & pr_{2N} \\ \vdots & \vdots & \vdots \\ pr_{N1} & pr_{N2} \cdots & pr_{NN} \end{pmatrix}$$

where each entry $pr_{i,j}$ is the probability of a person u that passed door d_i at time k, the probability of u to pass door d_j at time $k+1$. For each row i, $\sum_j pr_{ij} = 1$. From hereon, we denote the i-th row of Γ as $\Gamma(i, -)$.

A problem of using multinomial distribution is that the variance of the user movement is not controllable directly since the variance is determined by the numbers in $\Gamma(i, -)$ and the number of people that pass door d_i at time k, denoted as n_i. To model different movement variance, we assume that there exists a movement control parameter, denoted as ϕ. The motivation for introducing ϕ is that some people that move in indoor space (e.g. a shopping mall) will do random "wandering", which follows the walking patterns of the people around him; while others may have specific targets in mind and will walk based on the flow and the demographics. Effectively, ϕ controls the percentage of people who display "expected behavior" as defined below.

Movement Control Ratio: Specifically, assume that the number of people that pass door d_i at time k is n_i, then at time $k+1$, these $n_i * (1-\phi)$ will do deterministic walk based on $\Gamma(i, -)$, while $n_i * \phi$ people will do stochastic walk based on $\Gamma(i, -)$. More specifically, we split n_i people into two groups, one (expected, or deterministic) group with $n_i * (1-\phi)$ people, while the rest $n_i * \phi$ are in the "stochastic" group. The people in the deterministic group will walk to the next door in a deterministic fashion, i.e., $n_i * (1-\phi) * pr_{i,j}$ people will walk to door d_j for all j. For the "stochastic" group of people, we assume each person walks randomly based on the multinomial distribution given by $\Gamma(i, -)$.

Since the movement variance is determined by the number of people who do random walk, we can use ϕ to control this variance. When $\phi = 0$, movement variance is zero while the movement variance reaches its maximum when $\phi = 1$.

With all the assumptions aforementioned, our problem statement is summarized as follows:

Problem Statement: *Given a cell configuration graph with N sensors deployed, assume that at each time step k we have a set of noisy sensor readings $Y_k = [y_k^1, y_k^2, ..., y_k^N]$. Find out the distribution of people at each time step based on a set of sensor readings $Y_1, Y_2, ..., Y_T$ for T consecutive time steps, i.e., get a good estimate for the states X_k at each time step (we denote state estimates as $\hat{X}_k$).*

V. Theoretical Review of Particle Filters

Based on the aforementioned problem statement and those assumptions, we model the problem in the context of *dynamic systems*. A dynamic system has three components: the states, the measurement function and the system transition function. The states are X_k; the measurement function is presented in equation (1); the system transition function is presented in equation (2). The problem of tracking the distribution of people in indoor space becomes how to filter out the noise in Y_k to get an estimate, i.e., $\hat{X}_k$, for accurate state X_k in such a dynamic system characterized by equation (1) and equation (2) .

A common solution is to use the filtering methods. Kalman filters [12] work well when the system transition function is linear and measurement noise is Gaussian. When the system transition function is non-linear, an extended Kalman filter[13] method was proposed based on the first-order Taylor expansion. However, Kalman filters do not work well when the system is nonlinear or the noise is non-Gaussian. To bridge this gap, particle filters (a.k.a. Bayesian bootstrap, sequential Monte Carlo method) was proposed. It has been widely used in tracking, robot localization and many other applications due to its flexibility to deal with non-linear system with non-Gaussian measurement noise. Since our system transition function is non-linear, we decided to use the particle filtering method to filter out the noise.

The general idea of particle filters ([1], [6]) is to use a large number of particles (samples) to represent an unknown distribution, from which the samples are difficult to draw. The core part of particle filters is the *sequential importance sampling*(SIS) explained later.

The problem of getting a good $\hat{X}_k$ is equivalent to estimating the posterior distribution of the state X_k based on

TABLE I
SYMBOLS AND THEIR MEANING IN ALGORITHMS

T	total number of simulation steps
M	total number of particles
$\mathcal{P}_k$	$\mathcal{P}_k = \{P_k^1, P_k^2, ..., P_k^M\}, k \in [1, T]$
W	$W = \{w_1, w_2, ..., w_M\}$, where w_i is weight for particle P_i
Y	$Y = \{Y_0, Y_1, ..., Y_T\}$, where Y_k is sensor reading for time step k.
Γ	representing the *flow*, i.e., $p(X_{k+1}\|X_k)$.

all available measurements $Y_{0:k}$, i.e., $p(X_k|Y_{0:k})$. The key goal is to filter out the system noise and measurement noise. The idea of SIS is to first draw particles for X_{k+1} from the system transition function Γ given particles that represent the distribution of X_k. Then weights of the particles are updated. The weights of these particles quantify how much the particle is contributing to the targeted posterior distribution. By propagating the weights of the particles, a good estimate for the posterior distribution $p(X_k|Y_{0:k})$ can be reached. The sampling and weight updates for particles are two central parts in the particle filtering method. We elaborate on these two parts in Sections VI-B and VI-C.

VI. PARTICLE FILTERING BASED SOLUTION

Symbols and their meanings used in our algorithms are summarized in table I.

A. Overview of Particle Filtering Based Solution

Assume that for each time k we have an input of set $\mathcal{P}_k = \{P_k^1, P_k^2, ..., P_k^M\}$ of M particles and their weight vector W of size M, usually M is large. Each P_k^i is a vector of size N representing one particle. Initially, weights for particles are uniformly distributed. The pseudo code for the particle filtering (PF) based solution is presented in algorithm 1.

Algorithm 1 Particle Filtering Based Solution

Input:particles $\mathcal{P}$, and weights W, sensor readings Y, T.
Output: posterior distribution $p(X_{k+1}|X_k)$, i.e., estimates for X_k.
Assumption: Initially $w_i = 1/M$.
Method:

1: initialize particles by random sampling;
2: **for** ($k = 2$ to T) **do**
3: *Sequential Importance Sampling* M particles $P_k^1, ..., P_k^M$ from Γ (detailed in Section VI-B)
4: **for** i =1 to M **do**
5: update w_i to w_i' using Y_k and particles $P_k^1, ..., P_k^M$ (detailed in Section VI-C);
6: **end for**
7: normalize $w_1', w_2', ..., w_M'$;
8: compute $N_{eff} = \frac{1}{\sum_i w_i'}$
9: **if** ($N_{eff} \leq M$) **then**
10: *Resampling* for $P_k^1, ..., P_k^M$ and set $w_i' = 1/M$ (detailed in Section VI-D);
11: **end if**
12: get estimated $\hat{X}_k$ with weighted sum of $P_k^1, ..., P_k^M$ and $w_1', w_2', ..., w_M'$.
13: **end for**

Line 1 initializes the particles. We initialize the particle set $\mathcal{P}_1$ at time step 1 by random sampling utilizing the initial measurements and the noise model presented in equation (1). The reason is that although we do not know the initial accurate state X_0, we know the initial measurements Y_0. The idea is to take the measurements as if they are accurate, using the measurement model in equation (1) to sample values for each P_1^i, where $i \in [1, M]$. We contend that we can obtain a variety of good particles to start with by doing such a random sampling. Formally, if the measurement noise is drawn from a normal distribution, we initialize particles using equation (3):

$$P_1^i = Y_0 + \mathcal{M}_i, \tag{3}$$

where P_1^i is the i-th particle at step 1, Y_0 is the initial measurement vector, $\mathcal{M}_i$ is the noise vector based on the i-th particle at step one. Specifically, $\mathcal{M}_i = \{\mathcal{M}_i[j] | j \in [1, N]\}$ and $P_1^i[j] = Y_0[j] + \mathcal{M}_i[j]$ for all $j \in [1, N]$ and $i \in [1, M]$. Line 3 does SIS, Line 4 to Line 6 update the weights of particles, as explained later. A common problem that lies in particle filtering method is the "degeneracy" problem, i.e., after a few iterations, the weights of particles concentrate on a small portion of them while most of the particles have negligible weight. To avoid the "degeneracy" problem, line 8 is used to compute the degeneracy ratio, i.e., $N_{eff} = \frac{1}{\sum_i w_i'}$, if this ratio is smaller than the number of particles M, then *resmapling* (explained in VI-D) is used. The smaller N_{eff} is, the more likely "degeneracy" has occurred [1].

B. Sequential Importance Sampling

We now explain line 3: *sequential importance sampling*. SIS samples particles over the system transition function Γ. The input to the sampling part is a particle at time step k, i.e., P_k^i. The output is a particle for time $k+1$, i.e., P_{k+1}^i, where $i \in [1, M]$. The purpose of sampling is to simulate the walking process of people. Specifically, given the i-th particle P_k^i at time k, and the number of people that passed by door d_j, i.e., $P_k^i[j]$, how will these $P_k^i[j]$ people walk to $\langle d_1, d_2, ..., d_N \rangle$. For this purpose, we generate a walk matrix $W_{N \times N}$, each entry w_{d_j,d_l} indicates the number of people among the ones who walk through d_j at time k that will pass d_l at time $k+1$.

According to the movement model mentioned in section IV, given $P_k^i[j]$ people, $P_k^i[j]*(1-\phi)$ of them will do deterministic walk while $P_k^i[j] * \phi$ will do random walk based on Γ.

We obtain w_{d_j,d_l} using the movement model. Formally, for each door d_j, w_{d_j,d_l} is a sum of two separated parts: static part is computed using $P_k^i[j] * (1-\phi) * pr_{d_j,d_l}$; stochastic part is obtained by multinomial sampling. Pseudo code of the multinomial sampling part is presented in algorithm 2. By definition of Γ, each row j of Γ is actually a multinomial distribution for door d_j. We take a multinomial sample $\langle s_{j,1}, s_{j,2}, ..., s_{j,N} \rangle$ from this distribution with total number of trials equals $T_B = P_k^i[j] * \phi$.

The multinomial sampling is as follows: we sample $B(T_B, pr_{j,1})$ and get a number $s_{j,1}$; we then normalize the remaining probabilities and get $pr'_{j,2}, ..., pr'_{j,N}$. Next iteration, we sample on $B(T_B - s_{j,1}, pr'_{j,2})$ and get $s_{j,2}$. This loop terminates until we get $s_{j,N}$. This process is shown in algorithm 2 from line 5 to line 8.

Algorithm 2 Multinomial sampling component for SIS

Input: system transition function, i.e., the flow Γ , particle P_k^i, ϕ.
Output: multinomial part of $W_{N\times N}$, i.e., $W'_{N\times N}$.
Requirement: Sample drawing should be unbiased and non-deterministic.
Method:

1: initialize temporary matrix $W'_{N\times N}$;
2: **for** $j = 1$ to N **do**
3: get values $pr_{j,1}, pr_{j,2}, ..., pr_{j,N}$ from $\Gamma(j, -)$.
4: $T_B = P_k^i[j] * \phi$
5: **for** $l = 1$ to N **do**
6: draw a sample $s_{j,l}$ from the multinomial distribution using T_B and $\Gamma(j, -)$ by standard multinomial sampling;
7: $w'_{d_j,d_l} = s_{j,l}$;
8: **end for**
9: **end for**

The output of this multinomial sampling, i.e., $\langle s_{j,1}, s_{j,2}, ..., s_{j,N}\rangle$ is then added to entries from W_{d_j,d_1} to W_{d_j,d_N}. Based on $W_{N\times N}$, $P_{k+1}^i[j]$ is then the sum of row j in $W_{N\times N}$. Formally,

$$P_{k+1}^i[j] = \sum_l w_{d_j,d_l} \tag{4}$$

C. Update Weights for Particles

Now we explain the updates of weights from line 4 to line 6 of algorithm 1. Weight update is tricky and a theoretical background of the weight update can be found in an extended online version [1].

In theory, weights are updated through computing the probability $P(Y_k|P_k^i)$, which is the probability of how likely the measurement Y_k is from particle P_k^i. Our computation of $p(Y_k|P_k^i)$ is based on the measurement model. We know that Y_k is a vector of N noisy sensor readings. P_k^i is the i-th particle, i.e., an estimate for state X_k. Since noise for different sensors is independent, it is easy to see that $p(Y_k|P_k^i)$ can be computed using equation (5) according to the measurement model.

$$p(Y_k|P_k^i) = \Pi_j f(Y_k[j]) \tag{5}$$

where $f(Y_k[j])$ is the probability of $Y_k[j]$ on the normal distribution with mean equals 0 and standard deviation decided by $P_k^i[j]$. Since the probability of a given point on a continuous normal distribution is always 0, we used the Abramowitz and Stegun approximation to compute the probability, i.e., using the tail probability as the probability. The aforementioned sampling over system transition function and weight update comprise the core parts of SIS. Assume the dimensionality of a given CCG is N, i.e., the total number of sensored doors is N, then the time complexity for SIS is $O(M * N^2)$.

[1] http://cs.uvm.edu/~swang1/Indoor.pdf

D. Weighting and Resampling

We now explain Line 7 to Line 11 of algorithm 1. *Weighting* normalizes the weights of the particles, as described in line 7. The run time complexity of *weighting* is: $O(M)$. The motivation for the *resampling* ([5],[8],[14]) is to avoid the "degeneracy" problem known in the *particle filtering* method. To avoid this problem, at each iteration, we only select those particles that have higher weights. After resampling, weights are reset to uniform distribution, therefore, the discrepancy of weights between particles is minimized. The *resampling* process is just a standard way of sampling and is straightforward, we refer readers to [2] for details.

VII. Scalability and STAR model

A. Scalability of Particle Filters

The collapse of particle filters in large dimensions was thoroughly investigated in [19]. They provided theoretical proof that unless the number of particles grows super-exponentially to the system dimensionality, the particle filter method will collapse in large dimensionality. In our case, the dimensionality is the number of doors in a given CCG. However, since the time complexity of the particle filter method is $O(M * N^2)$, where N is the system dimensionality and M is the number of particles. If we increase particle size M to super-exponential order of system dimensionality N, the run time of the particle filters will be overwhelming. Therefore, we need to optimize the performance of particle filters under large dimensionality. The solution is to run the particle filtering method on disjoint subsets of the cell configuration graph.

An observation is that a real world floor plan can be partitioned into disjoint subgraphs with connecting doors. An example real world floor plan[2] is shown in figure 3(b). We can see that those offices are connected through the hallways and the offices located in the center can be viewed as a single disjoint subgraph. To formally model this property, we propose the STAR model for large indoor space.

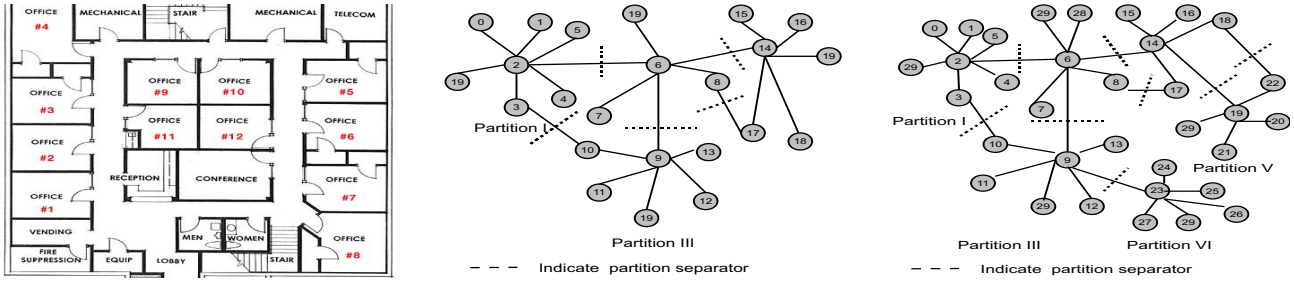
B. The STAR Model

A STAR model is defined as follows: there is a center node C with several other nodes c_i that are connected with C. Different parts of a building are connected together only through those center nodes C_is. There may also be very few connections between c_is from different subgraphs.

Formally, a STAR model can be defined as follows: a CCG that can be represented using a star model is a large graph $G_S = \{G_c^1 \cup G_c^2 \cup ... \cup G_c^L\}$, where each G_c^i is a star cell configuration. Figure 3(b) shows an example star CCG, where there are four star CCGs, with centers connected and a few non-center connections.

As another example, the cell plan in Figure 2 can be viewed as two subgraphs: one graph G_c^1 has cells c_0, c_1 and c_4; G_c^2 has c_3 and c_2; these two parts are connected through its centers c_0 and c_3, respectively. G_c^1 and G_c^2 are star cell configuration subgraphs. The STAR model provides a general guidance to

[2] http://www.strongofficesuites.com/images/second_floor.jpg



(a) An example real world floor plan (b) An example of STAR CCG (c) Large Scale CCG

Fig. 3. Example Real World Large Floor Plan, STAR CCG and Large Scale CCG

partition a large cell plan to several different smaller parts. This serves as the basis for us to apply particle filters method on large scale floor plans in order to track the distribution of people moving in large CCG.

Specifically, we first partition a large CCG into several star CCGs according to the STAR model, we then apply particle filters on each of these star CCGs and obtain an overall error estimate. For those doors that connect different star CCGs, we place *connecting doors sensor*, which has smaller noise than other doors. Intuitively, by applying this "partitioning" idea, we can guarantee a good local performance as well as a reasonably good overall performance. Meanwhile, we can reduce the running time of particle filters on large dimensionality compared to that of using super-exponential order of particles of N. Since now the time complexity will be $O(N^2 * M * L)$, where L is the number of subgraphs, and $L \ll N \ll M$. For those doors that connect different subgraphs, we call them connecting doors.

C. Measurement of Connectivity under STAR model

In this subsection, we define a measure to evaluate the inter-star CCG connectivity and intra-star CCG connectivity.

In order to measure the inter-star CCG connectivity, we adopt the connectivity measure used in [16]. For intra-star CCG connectivity, we define a new measure using the *algebraic connectivity* [15] and the *edge density* for graphs.

The inter-star CCG connectivity, denoted as δ, is defined as the ratio of inter-star CCG edges to the maximum number of inter-star CCG edges possible. Formally,

$$\delta(G|G_c^1, G_c^2, ..., G_c^L) = \frac{|\{v, u | v \in G_c^i, u \in G_c^j, i \neq j\}|}{n(n-1) - \sum_{l=1}^{L}(|c_l|(|c_l|-1))}, \quad (6)$$

where $|c_l|$ is number of nodes in star CCG G_c^l.

The intra-star CCG connectivity measure is defined in terms of *edge density* and *algebraic connectivity*. To introduce this measure, we first introduce *edge density* and *algebraic connectivity* .

The *edge density* of a star CCG G_c^l, denoted as ρ_l, is the ratio of the number of edges in that star CCG to the maximum possible number of edges in the same subgraph. Formally,

$$\rho_l = \frac{2 * |E_{G_c^l}|}{|c_l| * (|c_l| - 1)}. \quad (7)$$

The *edge density* ρ_l indicates how well the star cells are connected. Given two star CCGs, if the *edge density* is higher, then it means that the graph is better connected.

The *algebraic connectivity* of a graph, denoted as ψ is the second smallest eigenvalue of the Laplacian matrix induced from the graph. Specifically, the Laplacian matrix $\mathcal{L}$ of a given graph G is computed using $\mathcal{L} = D - A$, where D is the diagonal degree matrix of G, i.e., each entry i in the diagonal of D is the degree of node i; A is the adjacency matrix of G. The *algebraic connectivity* indicates the compactness of a graph. The *algebraic connectivity* is dependent on the number of vertices, as well as the way in which vertices are connected. The larger the *algebraic connectivity* is, the more compact a graph is.

To measure the intra-star CCG connectivity, we combine the *algebraic connectivity* and the *edge density*. Specifically, the intra-star CCG connectivity, denoted as η is measured as follows:

$$\eta = \Pi_{l=1}^{L} \rho_l * \psi_l \quad (8)$$

VIII. Performance Evaluation

In this section, we present our experimental design and performance analysis. The purpose of our experiment is to justify the effectiveness of our particle filtering based solution. We also want to find out how much sensor noise the particle filtering method can tolerate and how the movement variance, CCG scale and flow skewness affect the performance of the particle filtering based approach, respectively.

A. Data Generation

Data generation is based on input CCG. We generated our test data based on three different CCG. The smallest CCG has 5 cells with 8 doors, as shown in Figure 2; the medium CCG has 20 cells with 48 doors, as shown in Figure 3(b); the largest CCG has 30 cells with 72 doors, as shown in Figure 3(c). We randomly generated the flow for each CCG. For each flow, the probability of walking from d_i to its neighboring doors d_j are sampled from a multinomial distribution with its parameters sampled from a random uniform distribution.

Based on each flow, we generate one simulation data set with different movement variance parameter ϕ. During data generation, for each fixed movement variance parameter ϕ and a given door d_i, assuming the number of people that pass d_i is n_i, we let $n_i * (1 - \phi)$ people do deterministic walk while

TABLE II
PARAMETERS AND THEIR VALUES USED IN EXPERIMENT

# of Particles	250, 500, 750, **1000**, 1250, 1500,1750
ϕ	0.0, 0.1, 0.2, 0.3, **0.4**, 0.5, 0.6,0.7,0.8
# of People Large	160,320,480,640,**800**, 960,1120,1280,1440
# of People Medium	100, 200, 300, 400, **500**, 600, 700, 800,900
# of People Small	25, 50, 75, 100, **125**, 150, 175, 200,225
Measurement Noise	10%, 15%, 20%, **25%**, 30%, 35%, 40%
Connecting Door Noise	0%, 2.0%, **4.0%**, 6.0%, 8.0%

$n_i * \phi$ do random walk using the model described in section VI-B.

For each data set, we generate sensor readings for each door. We used normal distribution as our measurement model, i.e., the noise in the measurements are samples drawn from a zero mean normal distribution. Since we are concerned with the performance metric in equation (9), the standard deviation of the normal distribution is then a half-normal distribution. Specifically, assuming accurate reading for a given door d_i is n_i, sensor measurement noise level is σ_i, then $n_i * \sigma_i * \sqrt{\frac{\pi}{2.0}}$ is the standard deviation used in generating the measurements.

B. Experiment Setup

Parameters and their values are listed in table VIII-B with default values in bold. We set the sensor noise levels from 10% to 40% and use 25% as the default value. With higher cost and more intensive calibration, 10% accuracy is achievable as discussed in the related work. We use low-cost sensors, which can be very noisy. Our goal is to show that under high sensor noise, our method can still work reasonably well for distribution estimation.

In order to evaluate the performance of our particle filters based approach, we use the following performance metric proposed in [3]:

$$Err = \frac{1}{N}\sum_{i=1}^{N}\frac{|E_i - A_i|}{A_i} \tag{9}$$

where N is the total number of doors for a given CCG. E_i and A_i are the estimated and accurate value for d_i, respectively. If $A_i = 0$, then we use $|E_i - A_i|$ as the error for d_i. It is intuitive to see that if E_i is close to A_i, then *Err* is small.

For each parameter setting under a given flow, we change one of the parameter values and keep the others at default. All results reported are based on 10 runs for each parameter setting. For each fixed parameter setting and for each run, we have an averaged error avg_i and standard deviation $stddev_i$ over 100 simulation steps. So for 10 runs, we can get 10 avg_is and 10 $stddev_i$s. We then do average and standard deviation over these avg_i and $stddev_i$s to obtain averaged error rate and averaged standard deviations. Since the standard deviations $stddev_i$ are quite small, due to space limits, we skip reporting the figures for standard deviations.

For the purpose of experimental comparison, we also proposed a naive solution: it takes the measurements as if they are accurate.

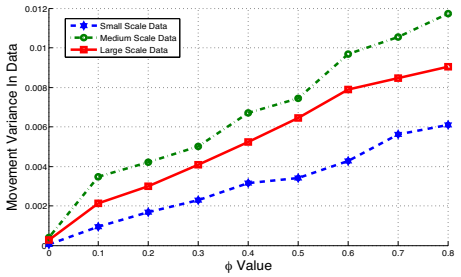


Fig. 4. Variance of Movement from Data

C. Result and Analysis

Variance of Movement from Data: Based on the generated data, we compute the real variance of movements as follows: assume that the number of people that pass d_i is n_i, the number of people that goes from d_i to d_j is n_{ij}, we compute the percentage of people that go from d_i to d_j as $\frac{n_{ij}}{n_i}$. Then for the door pair $\langle d_i, d_j\rangle$, we get a sequence of T percentages. We compute the variance of these T percentages, denote this variance as $\zeta_{i,j}$. The variance of movement for a given data set is then measured in terms of equation (10).

$$\gamma = \frac{1}{|\langle d_i, d_j\rangle|}\sum_{d_i,d_j}\zeta_{i,j}, \tag{10}$$

where $|\langle d_i, d_j\rangle|$ denotes the total number of $\langle d_i, d_j\rangle$ pairs in a given CCG. We plot the variance of movement for those three different CCGs based on different ϕ values, the result is presented in Figure 4. It is easy to see that the variance of movement from data generated in those three CCGs have the same trend. Specifically, when ϕ is larger, the variance of movement is larger. The reason is that larger ϕ means more people are doing random walk and fewer people are doing "expected" walk. The variance of the movement is then dominated by those people who do random walk, which has higher variance.

Connectivity measure of STAR model: Though the star CCGs can be obtained easily, we use the two measures in equations (6) and (8) to justify that obtaining star CCGs that conform with the underlying connectivity is the best partitioning strategy. For this purpose, we tried 10 different partitioning strategies that violate the underlying connectivity. We computed the intra-star CCG and inter-star CCG connectivity as well as error rate of these 10 testing cases. In each of these 10 testing cases, we keep the same number of particles, the same sensor measurement errors, the same number of better sensors. We only change the way that we partition the large CCG into different smaller CCGs instead of following the STAR model. Results for this testing are shown in Figure 5. We can observe from Figure 5 that the partition using the STAR model(namely, standard partition) has the smallest inter-star CCG connectivity and the largest intra-star CCG connectivity. Meanwhile, the standard partition has the smallest error rate among all 10 test cases. This justifies that partitioning strategy that conforms the STAR model is better than partitioning strategies that violate it.

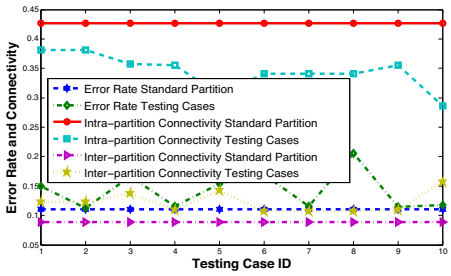


Fig. 5. Impact of Intra-Inter-Partition Connectivity

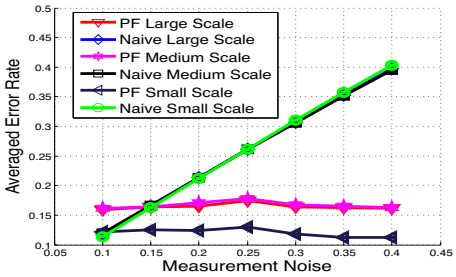


Fig. 6. Impact of Measurement Noise

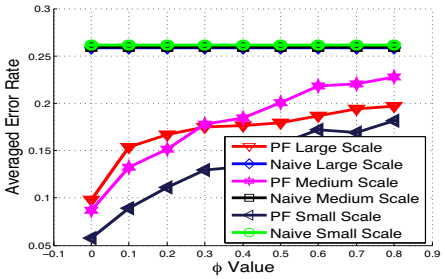


Fig. 7. Impact of ϕ

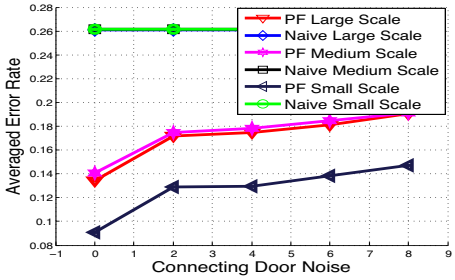


Fig. 8. Impact of Connect Door Noise

Impact of measurement noise: We can observe from Figure 6 that the error rate for naive method increases with the increment of measurement noise. This is intuitive since naive method takes the measurements as if they are accurate. If the measurement noise is high, the error rate is high. For our PF method, we can see that it outperforms naive method under all noise levels when scale is small and under most of the noise levels when scale is large. This shows that PF method can filter out noise in those sensor readings quite well.

Impact of ϕ: It is easy to see from Figure 7 that with the increase of ϕ value, the error rate of PF method increases since there are more people doing random walk. However, naive method's error rate do not change with different movement variance values because measurement has nothing to do with ϕ. Under all ϕ values, PF method works better than naive method. Meanwhile, if we look at Figure 7 and Figure 4 together, we can see that when ϕ is small, the variance of movement from the data is actually small, which is easier for the PF method to simulate the walk process. Therefore, the error rate for smaller ϕ will be better than that for larger ϕ values. This indicates that the mobility pattern of people will impact the performance of the PF method significantly.

Impact of connect door noise: From Figure 8, we can see that naive method's error rate does not change with respect to varying connecting door noise, because naive method uses the measurements only and measurement has no relation with connecting door noise. The error rate of PF method increases with larger connect noise, this is straightforward since for those connecting doors, we have more noisy readings. Therefore, the error should increase a little bit.

Impact of number of particles: The effect of particle size is presented in Figure 9. With more particles, the error rate for naive method do not change since the estimate from naive method is just the measurements. Measurements has no relationship with particle size. For PF method, the error rate fluctuates with number of particles since there are randomness in the method. In general, the error rate decreases with increment of number of particles. In theory, it is better to set larger number of particles for larger CCGs.

Impact of number of people: From Figure 10 we can see that the error rate of both naive method and PF method decreases with more people. This is because the indoor space becomes more crowded and it can tolerate more estimation errors. The degree of decrease on error rate for PF method is much larger than that of naive method. In short, PF and naive method favor more people.

Impact of CCG scale We can see from Figure 6 to Figure 10 that the CCG scale (i.e., number of doors) has important impact on the performance of both naive and PF method. For the impact of measurement noise, the performance of PF method increases with scale and measurement noise. This shows that in order for different scale CCGs to have same performance, we may need to set different number of particles in order to filter out more measurement noise. For impact of people, the performance of PF decrease under all scales we have tested, but the range that PF outperforms than naive method decreases with increment of scale, this means that for larger scale to work

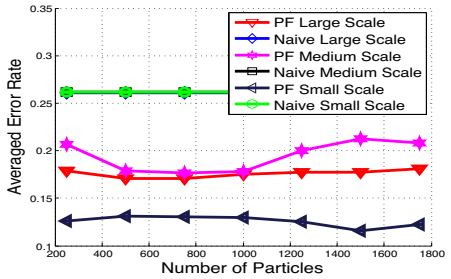
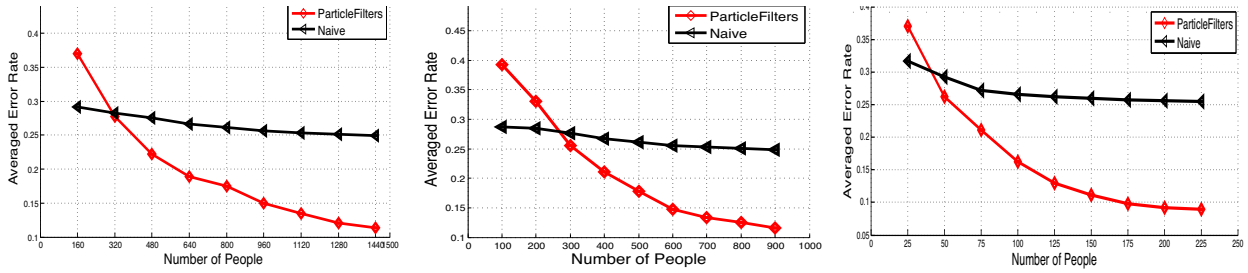


Fig. 9. Impact of Number of Particles



(a) Effect Of Number of People Large (b) Effect of Number of People Medium (c) Effect of Number of People Small

Fig. 10. Impact of Number of People

better, we need more people. This is true in real world situation since there are generally more people in a larger CCG. For impact of movement variance, the error rate of PF method increases with scale under the same number of people and the same number of particles. This is straightforward because larger CCG has larger dimensionality. Given the same number of people and particles, it is expected that the error rate will increase with scale. For impact of connecting door noise, for larger CCG, the number of connecting doors is larger since there are more subgraphs. With the increase of connecting door noise, the error rate for PF method will increase.

IX. Conclusion

In this work, we studied the problem of tracking distribution of people in indoor space with partitioning sensors. We first model the indoor space as a graph that contains *cells* and *doors*. Then we model the distribution tracking problem in the context of a dynamic system and proposed a particle filters based solution. To improve the performance of particle filters under large number of doors, we proposed a STAR model to partition a large indoor space into several smaller space and apply particle filters on each smaller space. Our particle filter based solution can reach a good estimate of the distribution of people under most of the cases.

However, in this work, we do not deal with a changing underlying "flow", i.e., we do not consider the temporal change in the "flow" model. We will investigate dynamic underlying flows in our future work.

Acknowlegement This research was partly supported by NSF grants CNS-0716575 and IIS-0415023. Work of Yan Huang was partially supported by the National Science Foundation Under Grant No. IIS-1017926. Work of X. Sean Wang was supported by the National Science Foundation, while working at the Foundation. Any opinion, finding, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

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, feb. 2002.

[2] M. Bolic, P. M. Djuric, and S. Hong. Resampling algorithms and architectures for distributed particle filters. *IEEE Transactions on Signal Processing*, 53(7):2442–2450, 2005.

[3] C.-Y. Chow, M. F. Mokbel, and T. He. Aggregate location monitoring for wireless sensor networks: A histogram-based approach. In *MDM '09: Proceedings of the 2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pages 82–91, Washington, DC, USA, 2009. IEEE Computer Society.

[4] G. Conrad and R. Johnsonbaugh. A real-time people counter. In *Proceedings of the 1994 ACM symposium on Applied computing*, SAC '94, pages 20–24, New York, NY, USA, 1994. ACM.

[5] R. Douc. Comparison of resampling schemes for particle filtering. In *In 4th International Symposium on Image and Signal Processing and Analysis (ISPA*, pages 64–69, 2005.

[6] N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107 –113, apr. 1993.

[7] K. Hashimoto, K. Morinaka, N. Yoshiike, C. Kawaguchi, and S. Matsueda. People count system using multi-sensing application. In *Solid State Sensors and Actuators, 1997. TRANSDUCERS '97 Chicago., 1997 International Conference on*, volume 2, pages 1291 –1294 vol.2, jun 1997.

[8] J. D. Hol, T. B. Schn, and F. Gustafsson. On resampling algorithms for particle filters. In *Nonlinear Statistical Signal Processing Workshop*, 2006.

[9] C. S. Jensen, H. Lu, and B. Y. 0002. Indoor - a new data management frontier. *IEEE Data Eng. Bull.*, 33(2):12–17, 2010.

[10] C. S. Jensen, H. Lu, and B. Yang. Graph model based indoor tracking. In *Mobile Data Management*, pages 122–131, 2009.

[11] C. S. Jensen, H. Lu, and B. Yang. Indexing the trajectories of moving objects in symbolic indoor space. In *SSTD*, pages 208–227, 2009.

[12] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[13] L. Ljung. Asymptotic behavior of the extended kalman filter as a parameter estimator for linear systems. *Automatic Control, IEEE Transactions on*, 24(1):36 – 50, feb 1979.

[14] S. Maskell and N. Gordon. A tutorial on particle filters for on-line nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2001.

[15] B. Mohar. The laplacian spectrum of graphs. In *Graph Theory, Combinatorics, and Applications*, pages 871–898. Wiley, 1991.

[16] S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[17] A. Schofield, T. Stonham, and P. Mehta. A ram based neural network approach to people counting. In *Image Processing and its Applications, 1995., Fifth International Conference on*, pages 652 –656, jul 1995.

[18] T. Teixeira and A. Savvides. Lightweight people counting and localizing in indoor spaces using camera sensor nodes. In *Distributed Smart Cameras, 2007. ICDSC '07. First ACM/IEEE International Conference on*, pages 36 –43, sept. 2007.

[19] B. L. Thomas Bengtsson, Peter Bickel. Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems. 2:316–334, 1 2008.

[20] B. Yang, H. Lu, and C. S. Jensen. Scalable continuous range monitoring of moving objects in symbolic indoor space. In *CIKM*, pages 671–680, 2009.

[21] B. Yang, H. Lu, and C. S. Jensen. Probabilistic threshold k nearest neighbor queries over moving objects in symbolic indoor space. In *EDBT*, pages 335–346, 2010.