Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

DATA ASSIMILATION WITH SENSOR-INFORMED RESAMPLING FOR BUILDING OCCUPANCY SIMULATION

Sanish Rai

Department of Computer Science and
Information Systems
West Virginia University Institute of Technology
410 Neville Street
Beckley, WV 25801, USA

Xiaolin Hu

Department of Computer Science

Georgia State University
25 Park Place
Atlanta, GA 30303, USA

ABSTRACT

A building occupancy simulation and estimation simulates the dynamics of occupants and estimates the real time spatial distribution of occupants in a building. It needs a simulation model and a data assimilation algorithm that assimilates real-time sensor data into the simulation model. In our previous works, we presented a graph-based agent-oriented simulation model, and a data assimilation framework based on Sequential Monte Carlo (SMC) methods for efficient real time occupancy estimation involving a large number of occupants. As the occupancy and the building environment size increases, there is a major problem with data assimilation caused by the high dimensional system states. To address this issue, this paper presents a new sensor-informed resampling method which utilizes sensor data to improve resampling of particles. Experiment results show the new method was able to provide better estimation of the system state with a limited number of particles compared to the standard bootstrap filter.

1 INTRODUCTION

A building occupancy simulation is used to study the dynamic spatial-temporal behavior and activities of occupants in buildings. It provides information like the number of occupants in an area, spatial distribution of occupants, activities happening in a space, and occupancy behavior during a certain period (Feng et al. 2015). Understanding the occupancy dynamics of a building is useful for various applications, including better evacuation for buildings with a large number of occupants, providing occupancy monitored energy service, design and control of smart environments, and crowd management in places such as airports and train stations. With the recent advances in sensor technology, most buildings are equipped with various kinds of sensors. They provide environment data and can be incorporated into simulation models for improved building occupancy simulation. This leads to more accurate simulation results and is especially useful for supporting real time decision-making tasks. Incorporating sensor data is achieved through data assimilation, which is the process of integrating observation data from the real world into a simulation model to produce accurate estimates of the system states. The estimated system state can then be used by simulation to generate improved simulation results. To implement data assimilation, both a simulation model that captures the state transition of the system and a data assimilation method that assimilates real time data into the model are needed.

To support data assimilation for building occupancy simulation, in previous work we developed a graph-based agent-oriented simulation model (Rai et al. 2015), and a data assimilation framework (Rai 2016) based on Sequential Monte Carlo (SMC) methods, also known as particle filters (Doucet et al. 2001).

The graph-based agent-oriented model combines the property of both graph based (Franz et al. 2005) and agent based models (Macal et al. 2009) to obtain their key advantages i.e. scalability and diversity. In this model, the graph-based feature means occupants' movement is modeled as flows from one node to another through a graph representing the building environment, and the agent-oriented feature refers to the model's capacity to consider agents as individual entities with unique features and decision making capabilities. The model has the advantage that it can capture the heterogeneity of individual occupants while in the meantime can simulate thousands of occupants in an efficient manner. This advantage is critical for supporting data assimilation for buildings with a large number of people like the game stadium and shopping malls. While the graph based agent-oriented model was able to model behaviors of building occupants, it lacks the analytic structures like those in equation-based numerical models. This makes it difficult to apply conventional estimation techniques such as Kalman filter and its variations. On the other hand, SMC methods are non-parametric filters are effective state estimation methods for systems with nonlinear non-Gaussian behavior. Thus we choose SMC methods to carry out data assimilation in our work. In SMC methods, the system state under estimation is represented by a large set of random samples, named particles, and their associated weights. These particles are propagated over time using importance sampling and resampling mechanisms.

While SMC methods enable data assimilation for building occupancy simulation using the graph-based agent-oriented model, several challenges exist for achieving an accurate estimation of the system state. A major challenge is caused by the high dimensional system states. As the size of the building structure increases, the number of nodes in the graph model increases. This increases the dimension of the system state, and thus require more particles in order to accurately estimate the system states. Hence the effectiveness of the data assimilation to represent the real states correctly decreases. And thus due to a large number of states, the SMC method suffers from the problems of sample impoverishment and particle deprivation. In sequential importance sampling, to avoid degeneration of particles, re-sampling is used to eliminate particles with lower weights and multiplying particles with higher weights. However, through re-sampling it is possible that after some runs of the data assimilation, all particles become identical. Thus there is sample impoverishment and the particle filter cannot represent all the diverse states of the real system. Particle deprivation happens when there is no particle near the true state. Consequently, it is impossible for particle filter algorithm to converge to the true posterior distribution of the target system. Particle deprivation problem can happen due to many reasons. One reason is the loss of diversity. Another reason is the increase of the dimension of the system states. The number of particles required to cover the state space grows exponentially with respect to the number of the dimensions of the system. Thus, if the dimension of the state space is highly complex, the limited number of particles cannot represent the true state and result in particle deprivation.

One way to improve the effectiveness of particle filtering is to develop more advanced sampling and/or resampling methods. In this paper, we focus on the resampling step of particle filtering and develop a new resampling named as sensor-informed resampling. The basic idea is to utilize sensor data to inform resampling of particles so that the resampled particles can better represent the true system state. The sensor data, although noisy, contains useful information about the true system state. As such, in cases where particles lack diversity and move further away from the true state, we can utilize the sensor data to improve the set of particles. In the standard particle filter resampling procedure, the entire system state is treated as a whole when being resampled. For data assimilation in building occupancy simulation, sensors are spatially located and each sensor reflects the state of its local region only. Thus we break the system state into smaller sub-states and carry out resampling at the sub-state level. We use the sensor data to generate sensor-informed sub-states, which are then used together with the particle filter sub-states during resampling. The resampled particles go through the sampling and weight calculation procedures in the same way as in the standard bootstrap filter. Using the sensor-informed resampling, it is possible to get better estimates of the occupancy than the standard bootstrap resampling only.

The remainder of this paper is organized as follows. Section 2 presents the related works. Section 3 briefly describes the graph based agent oriented model and the data assimilation framework that we developed and are used in this work. Section 4 presents the data assimilation based on the new sensor-informed resampling, and section 5 show some experiment results. Section 6 concludes this work and discusses future work.

2 RELATED WORKS

Building occupancy simulation is concerned with modeling and analysis of occupant property and behaviors. (Feng et al. 2015) have defined occupancy at four levels varying with time: the number of occupants in buildings, the occupancy status of space, the number of occupants in space and, the space location of occupants. Building occupancy finds its wide use in conserving resources such as (Maasoumy 2009) has developed a framework known as sensor utility network which uses information from various sensors to dynamically estimate the occupancy and conserve resources. Building occupancy provides a basis for a smart environment where we can simulate our test beds with various sensors and actuators. In their work (Cook et al. 2004) discuss the latest research in smart environment, philosophical and computational architecture considerations, network protocols, intelligent sensor networks and powerline control of devices, and action prediction and identification for the smart environment. A grid graph-based model known as ESM (Evacuation Simulation Model) is proposed by (Shen 2005) which considers the structural and spatial properties of the indoor space and shows advantages for indoor route analysis in evacuation simulation. The work builds a graph with each vertex representing a room, a segment of corridor or hallways and each edge representing a pipeline that occupant can transport on. The transportation of occupants on the edge of the graph is determined by factors such as social affiliation, access visibility, tenable time, speed, flow rate and the distance between rooms. (Tomastik et al. 2008)'s approach is based on three different models: an agent-based model, which includes the detailed description of each individual occupant's velocity, behavior and trajectory; a graph model, which represents the building structure and traffic dynamics using a graph, and a kinetic model, which models the congested areas of the building as queues to simulate the situation of congestion in the building.

The data assimilation algorithm used in this research is Particle Filters (PF). A framework of dynamic data driven simulation based on PF is presented by (Xue et al. 2012) for the forest fire spread simulation. In another research by (Gustafsson 2002), PF is used to develop a framework and algorithms to solve the problems of positioning, navigation, and tracking. The author presents a general algorithm based on marginalization enabling a Kalman filter to estimate all position derivatives. PF finds use in other fields like biology and chemistry as well. (Zhang et al. 2003) have used PF to create populations of compact long chain polymers and the relationships between packing density and chain length. PF are used by (Chen et al. 2008) to set up a probabilistic framework providing a basic for process fault diagnosis for the dynamic data rectification. (Wang et al. 2013) have presented a data assimilation framework for estimating movement patterns of about six occupants in an office environment. In our previous work (Rai et al. 2013), we developed a framework for using data assimilation for binary sensor data to predict occupancy behaviors in smart environments.

In our work, we modify the existing standard particle filter for improving the estimation of the states. In literature, we can find many variations of the particle filter to solve its two major problems, sample impoverishment, and particle deprivation. (Gilks et al. 2001) proposes a method to increase the diversity of the sample space by moving the particle around its original position. A re-sample move step is introduced at each iteration of the algorithm. It first samples new particles according to the weight of each of the particles and then moves the particles to some new positions in state space. () proposes a method by adjusting the size of particles utilized for estimation dynamically per Kullback-Leibler distance between the distribution estimated and the true distribution. The state space is divided into multiple bins and the particles that fall into a bin make it non-empty. After counting how many non-empty bins are there during estimation procedure, the number of the non-empty bin is used to calculate the desired number of samples using equation derived from Kullback-Leibler distance. (Doucet et al. 2000, Särkkä et al. 2007) have used Rao-Blackwellized particle filter to deal with the high dimensional state space of the target system. The state of each particle is divided into root nodes and leaf nodes which are linked using a dynamic Bayesian network. By doing this, the dimension of the system needs to be estimated reduces. (Kwok et al. 2005) proposes a new method based on Genetic Algorithm by treating each particle as chromosome and apply crossover and mutation operators. The crossover operation generates new chromosomes from existing chromosomes to increase the diversity of the sample space. (Bugallo et al. 2007) propose a method to deal with the high dimensionality of the state space by dividing the state space into subdimensions and apply a particle filter on each subdimension.

3 GRAPH BASED AGENT ORIENTED MODEL AND THE PF-BASED DATA ASSIMILATION FRAMEWORK

The graph based agent oriented model (Rai et al. 2015) is a low-resolution simulation model which describes the dynamics of the agents at a higher abstraction level. In the model, the structure of the building is represented using nodes and edges, and occupants are represented as agents with basic properties required for the movement. The movement of the occupants is modeled as a flow across the nodes through the links using queuing and flow theory. Since movements of occupants in the network are more orderly organized, occupant behaviors at higher resolution are not necessarily required for the simulation. The occupancy behaviors in situations like crowd and congestion can be well represented by lower resolution models like graph models. This significantly reduces the computational complexity of the simulation without losing the overall property of the occupancy behavior. Figure 1a represents the framework for a graph based agent oriented model which show the agents and graph model. The graph model consists of the queues and the building structure is managed by nodes and links. The agents are provided with decision making capabilities which are governed by the rule engine maintained by the user. We also provide a GUI based interface where the occupant's behavior can be observed in a building environment.

More specifically, the building environment is represented using vertex for nodes and connecting structures like doors, stairs by links. The occupant in the model is represented using a variable o having state s, where

$$o = < ph, v_{position}, id, g_{id}, l_{from}, s >$$

and,
$$s = \begin{cases} staying \\ moving \\ inqueue \end{cases}$$

Here, ph represents the phase of the occupant which represents the environment condition of the building, $v_{position}$ represents the current vertex the occupant reside in, id represents the identification of the occupant, l_{from} represents the link that the occupant come from, s represents the state of the occupants and has three different values, *staying*, *moving* and *inqueue*. The occupant has three basic states, *staying* to represent that the occupant is staying in a node, *moving* to represent that the occupant is moving across the node and *inqueue* to represent that the occupant is waiting to enter a node. To enter a node, the occupants follow the basic queue behavior. Each link has a flow rate and occupants enter based on the flow rate. If the node is full or the flow rate is low due to high density at the queue, occupants will wait in the queue to enter the node.

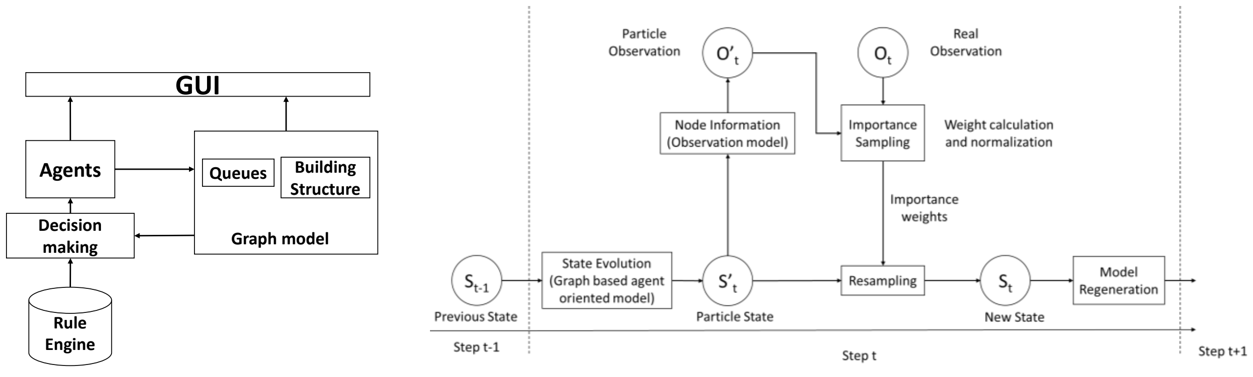

Figure 1: a) Graph based agent oriented framework b) PF-based data assimilation framework

To support data assimilation of the graph-based agent-oriented model, we used the basic bootstrap filter as the algorithm for the data assimilation (Rai 2016). The observations are the occupancy information which is collected by various sensors installed in the rooms. In our framework, we consider only the node information as the state which reduces the size and complexity of the state for data assimilation. Each of the particles goes through the system transition function which evolves the system from the current system to a new state $S_{t+1} = G\ (S_t) + Q_t$. Here, G represents the state transition function based on graph based agent oriented model and Q_t is the system processing noise. The observation data for the particle filter is represented as $O_n = Obs(n) + m_t$. Here, $Obs(n)$ is the observation at room n and m_t is the measurement noise for the observation. Figure 1b represents the PF based data assimilation framework in which S_{t-1} is the previous state of the system which is evolved by the state transition function to new state S'_t which provides its observation (sensor data). This observation is compared with the real observation and weight computation and normalization are done in the importance sampling step. In the resampling step, particles are resampled based on the weights and the new state is estimated which is closer to the real system state. After that, there is a model regeneration step, which generates the required necessary properties of the system from the resampled state like agent and queue properties. The algorithmic structure of the standard particle filter with resampling is as follows:

Sequential importance resampling PF
Input: Occupancy states and corresponding importance weights at step t $\{S^i_t, w^i_t\}$
Output: Occupancy states and corresponding importance weights at step $t+1$ $\{S^i_{t+1}, w^i_{t+1}\}$

```
1. Sampling
   For each state $S^i_t$, draw a sample $S^i_{t+1}$ from $p(S^i_t \mid S^i_{t+1})$ using $G(S^i_t)+Q_t$
2. Weight Updating
   For each state, update the weights $w^i_{t+1} = w^i_t \times p(O_{t+1}|S^i_{t+1})$
   Normalize $w^i_{t+1} = \frac{w^i_{t+1}}{\sum_{i=1}^{N} w^i_{t+1}}$
3. Resampling
       Draw $N$ particles from $S^i_{t+1}$ based on $w^i_{t+1}$
```

4 DATA ASSIMILATION USING SENSOR -INFORMED RESAMPLING

The high dimensionality of the system increases the challenges of sample impoverishment and particle deprivation in the PF based data assimilation framework. The building occupancy simulation consists of a multidimensional state space where the nodes and the agent properties represent dimensions of the system. As such, when the state of the system increases, the number of particles required to estimate true states grows exponentially. This leads to a higher computation cost for simulating building occupancy for a large number of occupants in big environments. We present a new resampling method to utilize the available sensor data to deal with the issues introduced by the high dimensionality of the system.

The goal of sensor-informed resampling is to utilize sensor information to generate samples that are closer to the real system state. To achieve this we modify the standard resampling step, which generates a new set of particles according to the importance weights of particles, by incorporating information from sensors. The basic idea is that if a sensor data has a high value in terms of a number of occupants, then the real system is likely to have a large number of occupants in the node where the sensor locates, and this information can be used to adjust the particles generated from the standard resampling step. To do this in a systematic way, there are two issues that need to be resolved: 1) how to extract the most information from a sensor data in order to support data assimilation; 2) how to incorporate the sensor information into the standard resampling step. The first issue is important because typically there are a limited number of sensor data points, i.e., sensors are sparsely deployed in the building. Nevertheless, the rooms in a building are related to each other due to their spatial locations. When one room is congested, there is high chance that its neighboring rooms also have a high number of occupants. Thus the sensor data collected from one room can also provide information about its neighboring rooms, and this information can be used to improve the resampling step of the particle filter. Below we describe more details regarding how we address these two issues.

4.1 Extract information from sensor data

We can use sensor data to obtain information not only for the nodes where there are sensors but also for the neighboring nodes. This is because, in a building, the occupancy flow is directly related to the connectivity of the nodes. An occupant can move only to a connected node and to reach a destination, they need to move through the connected nodes from a source to destination (Hallways, passages, stairs, elevators etc. are also represented as nodes). As such, based on the occupancy of a node, we can get an idea of the occupancy in its neighboring nodes. To give an example, in Figure 2: , if Node 13 and Node 15 has high occupancy, then most probably Node 14 has high occupancy. But if Node 13 has high occupancy and Node 15 has few, two cases may arise: first Node 14 may be the main destination and have high occupancy; second, Node 13 may be main destination and Node 14 may not have high occupancy. For the total occupants, if there are few occupants in other sensors, Node 14 is the main destination, otherwise, if other sensors have a significant number of occupants, Node 13 is the main destination. Here, we narrowed down our estimation to two possible cases, after which we can use the data assimilation framework to help estimate one of the cases.

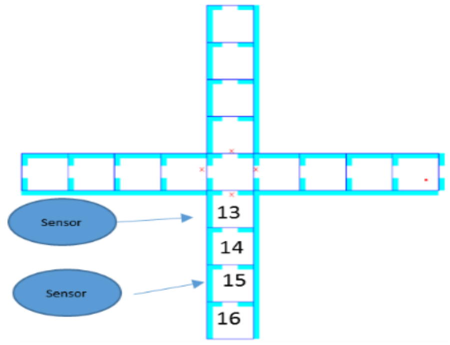

Figure 2: Issue caused by placement of sensors

There can be various methods for generating estimates for nodes based on sensor data from some neighboring nodes. Here we describe one basic approach for layout in which nodes are connected in a single direction. We illustrate this approach using the example shown in Figure 3. In Figure 3a, let us assume that G, F, E, X, A, B, C are six connecting nodes and only node X has a sensor. Based on the location of the nodes, we can use the sensor from node X to get an estimate of nearby nodes. During the movement of occupants, when node X has a high occupancy, the occupancy will decrease as the distance of nodes increase from node X. Thus, we can assume that the occupancy will be somewhat distributed following spatial Gaussian distribution across the nodes. The distribution is centered at node X, and spreads to farther nodes. In figure 3b, we can see an example of the distribution where X-axis represents the nodes based on their id and Y-axis represents the occupancy which follows the spatial Gaussian distribution. Here, X_0 represents the sensor data in node X. From the distribution, we can get an estimate of occupancy for node B and E to use in resampling by randomly taking a normal distribution between X_0 and X_1. Similarly, for node C and F between X_1 and X_2 and for node D and G between X_2 and X_3. In some cases, the nearby nodes may have higher occupancy as well. To consider for such cases, for each node we will add noise to the estimated occupancy.

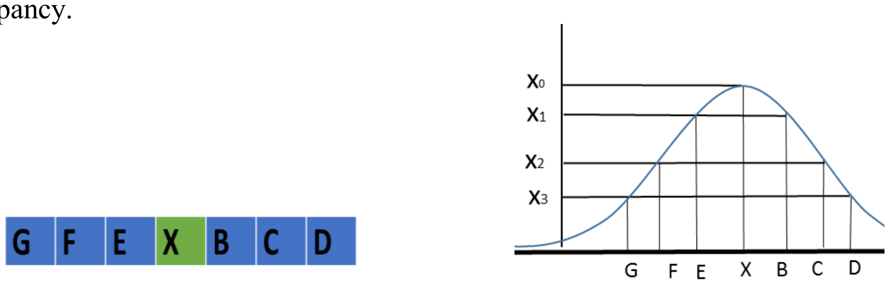

Figure 3: a) Layout of connecting nodes b) spatial Gaussian distribution

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\sigma^2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{1}$$

The probability density of the spatial Gaussian distributions is represented by (1). Here, μ represents the node with the sensor data and X represents the neighboring nodes. σ represents the location of the neighboring nodes based on their location from μ. Here we observe that when σ is 0, it represents the sensor data for the node with the sensor. If σ is 1, we can obtain information for two neighboring nodes E and B with a high probability density. However as we increase σ, the probability density is lower and so the estimation may not be very accurate. Thus we observe that in this method, we can obtain information about the adjacent neighboring node with higher confidence than others. We will use information about only the adjacent nodes in this work. In case there is a node which has two adjacent nodes with the sensor, using information from all the nodes will help get a better estimate of occupancy in the node. In such case, we can use some method of combining both information to get a better estimate. In this work, we will use the average of the estimates to use as an occupancy in the node and then will use it in the resampling method.

4.2 Sensor-informed resampling

To use sensor data for improving the convergence of the particles, we will generate estimates of some of the sub-states from the sensor data and then combine them with the existing particles of the standard particle filter. In the standard particle filter resampling, we treat the entire system state as one system and so, in building occupancy a particle consists of various nodes with occupancy information. We can divide the whole system state into a set of sub-states where each sub-state is for a subgraph of the overall graph model (i.e., the overall building layout). These sub-states, when brought together, represents the overall state of the system. Each sub-states consists of individual nodes with state variables like occupancy count, occupancy behavior, and other node properties. Since we can use sensor data to generate estimates of the node with the sensor and other adjacent nodes without the sensor, we can create sub-state estimates of these nodes. This sub-state estimates can now be placed in the particle and this new sensor-informed particle will be able to estimate the real system with improved accuracy. To consider the noise in the sensor, we will replace only a randomly selected subset of the particles and the randomly selected sub-states in each particle. Thus, the process will impact only some of the particles to improve the diversity of the system state that the particles represent. Since we will not modify all of the particles, some of the good original particles are preserved. And also if the sensor data is faulty, it will get filtered out during the next iteration of data assimilation along with the particle.

Using the real sensor data, we will compute the sub-state for nodes with and without sensors using the spatial Gaussian distribution described in Section 4.1. The sub-states consists of the number of occupants in the node and their direction. We will compute more than one set of sub-states to provide multiple estimates from a sensor data for the same node. During the resampling step of the particle filter, after resampling we select a few of the standard particles and mix sub-states generated from sensor data. We follow rest of the standard resampling procedure from the bootstrap filter. We will set the number of occupants in each node as the sub-state and we will generate other properties of the nodes using the data assimilation model. Formally, let us have a set S_t of n particles randomly selected from the original set of particles. Here each particle has m sub-states,

$$S_n = <D_1, D_2, \ldots\ldots D_n> \textit{ and}$$

$$D_n = <N_1, N_2 \ldots\ldots N_m>$$

where D_n represents a particle and N_m represents a sub-state. Let X represent the set representing p sets of estimates from sensor data. Each set of estimate will contain q sub-states,

$$X = < I'_1, I'_{2,\ldots.}\ I'_p >, \textit{ and}$$

$$I'_p = <N'_1, N'_{2,\ldots.}\ N'_q >$$

Here, q depends on the number of nodes with sensor since we can get information only for the adjacent nodes. With these set of particles and sub-state estimates, a new particle is constructed as

$$\boldsymbol{\widetilde{D}} =< \boldsymbol{\widetilde{N_1}}, \boldsymbol{\widetilde{N_2}} \ldots \ldots \boldsymbol{\widetilde{N_m}} >$$

where $\widetilde{N_m}$ represents the sub-state selected randomly from one of X. For each particle , we first randomly select a set of sub-state from X and then replace sub-states from I'_p. We will use a probability for replacing a sub-state so that all of the sub-states are not replaced.

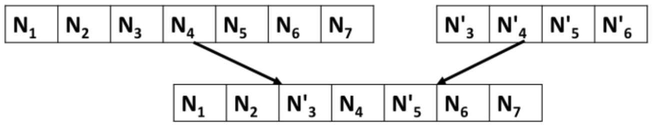

Figure 5: Example of sensor-informed resampling

Figure 5 gives an example of sensor-informed resampling for a particle. In this example, the particle has seven sub-states. We use a set of sub-state generated from available sensor data which has information for node 3, 4, 5 and 6 only. After replacing the sub-states we observe that only sub-state information from node 3 and 5 is replaced in the original particle. Similarly, a different replacement will occur for another particle in which the sub-states from the sensor data will be another value from state Gaussian distribution. These sensor-informed particles now go through the steps of resampling and have a better probability of representing the real system. We can define the algorithm for resampling using the new method as below.

Algorithm for sensor-informed resampling

```
    Resampling step:
    Input: The set of $N$ particles with associated weights
    Set $I$ with sub-states generated from sensor data
```

```
Output: A new set of sensor-informed N particles

Perform the standard resampling for particles
Select k particles from original particle
For each selected particle
        For every sub-state from I,
            Incorporate sub-state with particle
        End for
End for
```

5 EXPERIMENT

In this section, we conduct an experiment using particles from sensor data and compare the results with the standard resampling. We want to experiment using a large number of occupants but without increasing the number of particles highly. Thus, we want to maintain the computational cost of the simulation for large occupancy. To verify it, we set the capacity of the room to 200, the length of the edge to 35m and the number of occupants to about 16,000 so that the complexity of the model is high and it becomes difficult for the general data assimilation to detect the occupancy behavior. The task of estimating an exact number of occupants is highly difficult at such complex system state hence we focus on detecting congestion i.e. when there are a very high number of occupants in a node. For this, we can define a threshold value which defines the minimum number of occupants to create a congestion situation. When a node contains more than that number of occupants, it is in a congestion state. Figure 6 shows the building structure for the simulation. There are 79 nodes and the structure mimics an airport/train terminal type of environment where people enter the system through one or more entrances (Node 0 and 29); then they move to one of the terminals for boarding. There is only one door connecting two nodes and occupants will need to choose that door to go to the connecting node. We set up sensors by placing one node without sensor between two nodes with the sensor. Using the identical twin experiment, we create a real system with congestion in two nodes (Node 113 and 195) for some period and observe the congestion which propagates across the neighboring nodes.

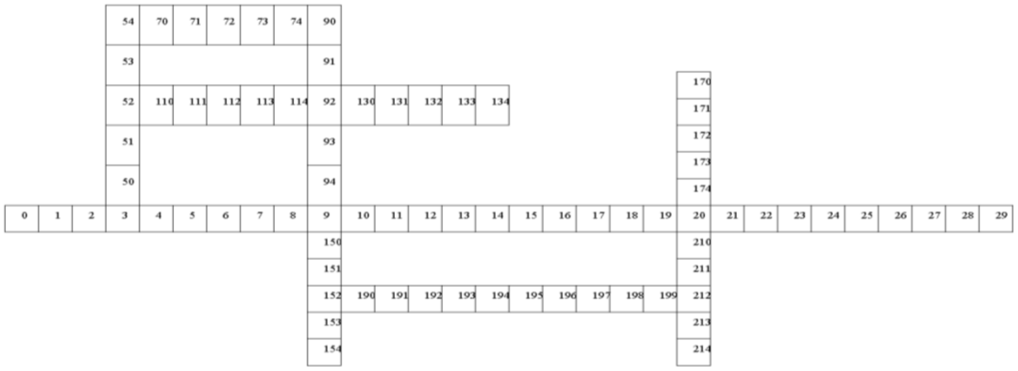

Figure 6: Building structure for simulation

Figure 7 shows the result of comparison where blue graph represents the occupancy from a real system, orange graph from the bootstrap PF (standard resampling) and red graph from the resampling using sensor-informed PF. For convenience, we compare the results of the main destination nodes and their neighboring nodes. The peak of the blue graph shows the maximum capacity of the node and we try to compare how quickly and closely the data assimilation can estimate it. We observe that the bootstrap particle filter is not able to detect congestion in real time compared with the sensor-informed PF. We create a 1/5th of the total standard particles as a set of particles using sensor data and randomly select the sub-states from both sets of particles. We see that the bootstrap PF is quite slow to detect congestion in the neighboring nodes. However, the sensor-informed particle filter can predict congestion (occupants more than half the capacity) at the same time as the real system. In node 114 we observe that the standard PF estimates the peak occupancy around timestep 120. It is due to the reason that the occupants keep accumulating in the room and do not leave due to limitations in estimating by the bootstrap PF.

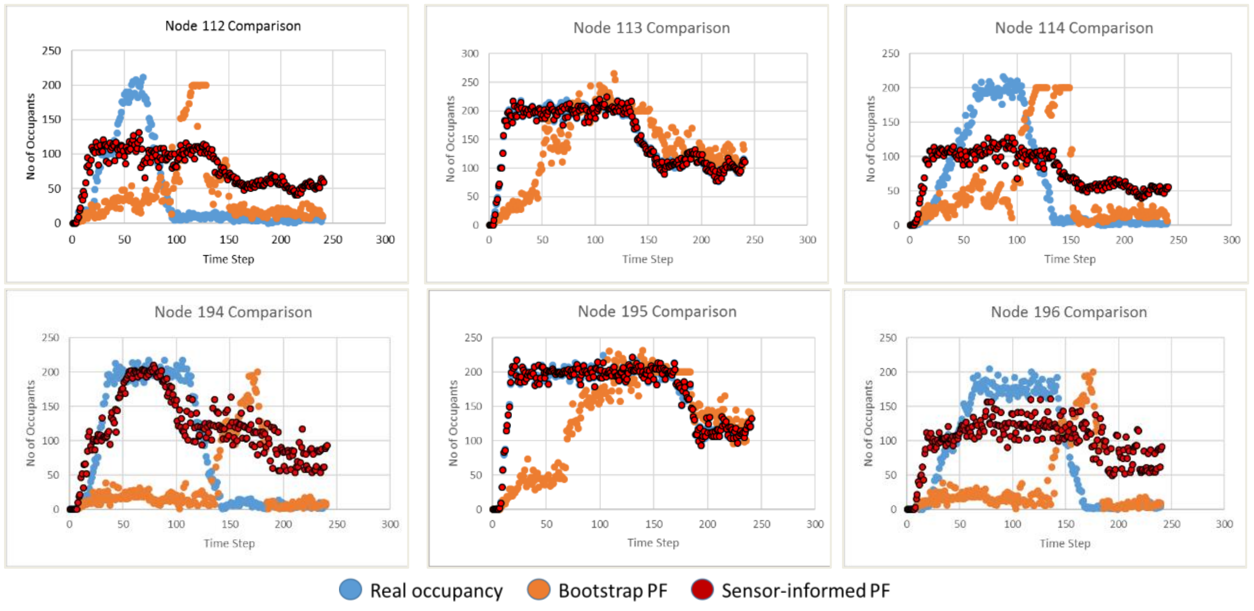

Figure 7: Comparison using sensor-informed PF with 200 particles

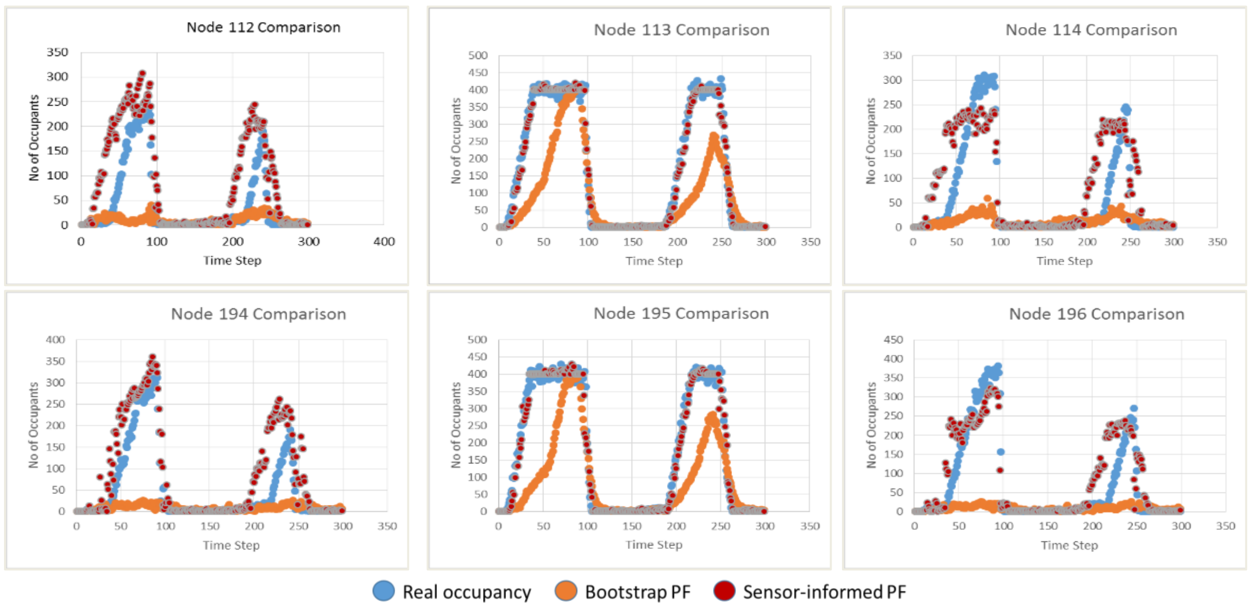

Figure 8: Comparison using sensor-informed PF with 100 particles

In the second experiment, we increase the size of the room to 400, edge size to 35 and use about 9500 agents for a congestion simulation. Congestion is created in nodes 113 and 195 in two stages and we use only 100 particles to detect the occupancy. Since the number of particles is quite a few, the occupancy is difficult to estimate using the standard bootstrap particle filter. However, using our new algorithm we are able to estimate the occupancy pattern for the congestion with the limited number of particles. Figure 8 shows the result of the comparison, here also blue graph is the occupancy from real simulation, the orange graph is the simulation using the bootstrap PF and red graph is the result of sensor-informed PF. We observe the result of sensor-informed PF is much better than bootstrap PF. The bootstrap PF is able to make some estimation for the main destination nodes but is not able to estimate the occupancy for the neighboring nodes. However, the sensor-informed PF can make a good estimation for neighboring nodes as well. As such, we see that the new algorithm makes good predictions utilizing the sensor data and use fewer particles compared with the standard bootstrap particle filter.

6 CONCLUSION

In this paper, we presented a new resampling named as sensor-informed resampling which can estimate the occupancy better than the standard resampling method. The method utilizes sensor data to inform resampling of particles so that the particles can better represent the true system state. The improved resampling helps in solving the major issues caused due to increase in complexity of system state when occupancy and environment size increase. From the experiments, we observed that the sensor-informed resampling is better in estimating congestions than the standard particle filter when there is high occupancy. In future, we plan to work on analyzing the output of the algorithm when we introduce more noise. We also plan to implement other advanced methods of utilizing sensor data during the resampling step.

REFERENCES

Feng, X., Yan, D. and Hong, T. 2015. “Simulation of occupancy in buildings”. *Energy and Buildings*, 87, pp.348-359.

Rai, S., Wang, M., and Hu, X. 2015. “A graph-based agent-oriented model for building occupancy simulation”. *In Proceedings of the Symposium on Agent-Directed Simulation,* pp. 76-83. Society for Computer Simulation International.

Rai, S. 2016. “Building occupancy simulation and data assimilation using a graph based agent oriented model”. *Dissertation, Georgia State University,* 2016. http://scholarworks.gsu.edu/cs_diss/114

Doucet, A., De Freitas, N. and Gordon, N. 2001. “An introduction to sequential Monte Carlo methods.” In *Sequential Monte Carlo methods in practice (pp. 3-14).* Springer New York.

Franz, G., Mallot, H., Wiener, J. and Neurowissenschaft, K. 2005. “Graph-based models of space in architecture and cognitive science-a comparative analysis”. In *Proceedings of the 17th International Conference on Systems Research, Informatics and Cybernetics* (Vol. 3038).

Macal, C.M. and North, M.J. 2013. “Agent-based modeling and simulation: introductory tutorial”. In *Proceedings of the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World* (pp. 362-376). IEEE Press.

Rai, S. and Hu, X. 2013. “Behavior pattern detection for data assimilation in the agent-based simulation of smart environments”. In *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*-Volume 02 (pp. 171-178).

Maasoumy, M. 2009. “Estimation of Occupancy Distribution in Buildings”. *Univ. Calif. Berkeley*.

Cook, D. and Das, S.K., 2004. *Smart environments: Technology, protocols, and applications* (Vol. 43). John Wiley & Sons.

Shen, T.S. 2005. “ESM: a building evacuation simulation model”. *Building and Environment*, 40(5), pp.671-680.

Tomastik, R., Narayanan, S., Banaszuk, A. and Meyn, S. 2010. “Model-based real-time estimation of building occupancy during emergency egress”. In *Pedestrian and Evacuation Dynamics* 2008 (pp. 215-224). Springer Berlin Heidelberg.

Xue, H., Gu, F. and Hu, X. 2012. “Data assimilation using sequential monte carlo methods in wildfire spread simulation”. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 22(4), p.23.

Gustafsson, F., Gunnarsson, F., Bergman, N., Forssell, U., Jansson, J., Karlsson, R. and Nordlund, P.J. 2002. “Particle filters for positioning, navigation, and tracking”. *IEEE Transactions on signal processing*, 50(2), pp.425-437.

Zhang, J., Chen, R., Tang, C. and Liang, J. 2003. “Origin of scaling behavior of protein packing density: A sequential Monte Carlo study of compact long chain polymers”. *The Journal of chemical physics*, 118(13), pp.6102-6109.

Chen, T., Morris, J. and Martin, E. 2008. “Dynamic data rectification using particle filters”. *Computers & Chemical Engineering*, 32(3), pp.451-462.

Wang, M. and Hu, X. 2013. “Data assimilation in agent based simulation of smart environment”. In *Proceedings of the 1st ACM SIGSIM Conference on Principles of Advanced Discrete Simulation* (pp. 379-384). ACM.

Gilks, W.R., and Berzuini, C. 2001. “Following a moving target—Monte Carlo inference for dynamic Bayesian models”. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(1), pp.127-146.

Fox, D. 2001. “KLD-sampling: Adaptive particle filters and mobile robot localization.” *Advances in Neural Information Processing Systems (NIPS)* 14.1, 26-32.

Särkkä, S., Vehtari, A. and Lampinen, J. 2007. “Rao-Blackwellized particle filter for multiple target tracking”. *Information Fusion*, 8(1), pp.2-15.

Doucet, A., De Freitas, N., Murphy, K. and Russell, S. 2000. “Rao-Blackwellised particle filtering for dynamic Bayesian networks”. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence* (pp. 176-183). Morgan Kaufmann Publishers Inc..

Kwok, N.M., Fang, G. and Zhou, W. 2005. “Evolutionary particle filter: re-sampling from the genetic algorithm perspective”. In *Intelligent Robots and Systems,* 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on (pp. 2935-2940). IEEE.

Bugallo, M.F.; Ting Lu; Djuric, P.M. 2007. "Target Tracking by Multiple Particle Filtering," *Aerospace Conference,* IEEE, pp.1,7.

AUTHOR BIOGRAPHIES

SANISH RAI is an Assistant Professor in the Department of Computer Science and Information Systems at West Virginia University Institute of Technology, Beckley, WV. He received his Ph.D. degree from Georgia State University in 2016. His research interests include simulation and modeling, agent and graph based systems, and data assimilation. His email address is sanishrai@gmail.com.

XIAOLIN HU is an Associate Professor in the Department of Computer Science at Georgia State University, Atlanta, Georgia. He received his Ph.D. degree from the University of Arizona in 2004. His research interests include modeling and simulation theory and application, agent and multi-agent systems, and complex systems science. His email address is xhu@cs.gsu.edu and his web page is https://grid.cs.gsu.edu/xhu/.