Simulating Events to Generate Synthetic Data for Pervasive Spaces

A. Mendez-Vazquez, A. (Sumi) Helal, and D.J. Cook

***Abstract*—a constant problem in the generation of data for testing pervasive spaces is the lack of real standards in the data generation, and the cost of the pervasive space itself. This hinders the development of the pervasive spaces. If pervasive systems is going to be included into a particular mass production device, for example cars, you should be able to give a percentage of how secure is your system. For example, testing the robustness of the new artificial intelligence layers to increase the capabilities of the pervasive spaces will require large amounts of data portraying different scenarios. Then, it is necessary to design realistic simulations of the pervasive spaces to study the possible advantages and limitations of particular collections of actuators, sensors and systems. A first steep in this direction is trying to simulate chain of events from the sensor output space by the use of pattern recognition models. Specifically, we propose the use of Markov Chains to generate patterns of events due to the fact that they have been used successfully to classify chains of events in machine learning and computer vision. In addition, we propose the use of Poisson spike generators for the randomization of the time stamps for these events. Finally, we use a series of distributions over the ranges of each sensor to obtain a realistic spread on the sensor output values. We believe these strategies will make for a more realistic simulation of the space of sensors in a pervasive space.**

***Index Terms*—pervasive spaces, simulation, Markov chains, state machines, spike Poisson generator.**

I. Introduction

Actual data collected from real deployments is ultimately the best data that can be had and used in evaluating systems and new concepts. However, available data may not be specific enough to drive certain evaluation goals. Therefore, it is necessary to propose, as an alternative, a method to generate synthetic data for pervasive spaces.

In recent years it has become obvious that the increasing costs of building pervasive spaces without a correct design and planning makes the research in this area extremely difficult [1]. It is more not everybody has a large budget to build a pervasive space to test new algorithms and ideas. In addition, if you still have the budget to build a pervasive space, it can be time prohibitive to generate enough data for a meaningful collection of patterns or events. This makes necessary to look for ways of creating realistic simulations [8][9] of events to fine tune the analysis and understanding of the pervasive space to be created. This early stage simulation can really help to understand all the hurdles involved in the design and construction of a pervasive space. It is more, simulation is at the core of any successful industrial design because before any concept enters into production, an extensive simulation and prototyping is developed to asses their feasibility.

If pervasive spaces are going to be something more than a simple academic exercise, it is necessary to focus into the development of accurate and realistic models to simulate and generate data. This will allow us to test new ideas from the realm of artificial intelligence and computer vision to add the sought flexibility and adaptability that many researches are looking for. A first step in this direction is trying to simulate a chain of events in the sensor output space by the use of pattern recognition models. Specifically, we propose the use of Markov Chains to generate patterns of events due to the fact that they have been used successfully to classify chains of events in machine learning and computer vision. We believe this strategy will make for a more realistic simulation of sensor activity in a pervasive space.

In this paper, we propose to use of a combination of Markov chains [2][11], Poisson processes [3][12] and probability distributions [4] for the simulation and generation of synthetic data for pervasive spaces. Initially, The Markov Chain will be used to generate pattern of activities based in *apriori* knowledge of the people behavior. This is due to the fact that Markov chains have been successfully used in the detection of pattern of activities [5][6][10][13][14][15][16]. Therefore, it is natural to use them in the inverse problem i.e. instead of

Andres Mendez-Vazquez is with the Department of Computer and Information Science and Engineering, University of Florida, -P.O. Box 116120, Gainesville, FL 32611-6120 USA. Email: amendez-@cise.ufl.edu.

Abdelsalam (Sumi) Helal is a Professor is with the Department of Computer and Information Science and Engineering, University of Florida, -P.O. Box 116120, Gainesville, FL 32611-6120 USA. Email: helal@cise.ufl.edu.

Diane J. Cook is with the School of Engineering and Computer Science, Washington State University, EME 121 Spokane Street Box 642752, Pullman, WA 99164-2752, USA. Email: cook@eecs.wsu.edu

discovering patterns of activities; we use them to generate those patterns. Now, we need a way to generate random time stamps in a given time interval for each event or sensor output to be simulated. A popular way to generate time stamps in a given time interval for events is by the use of the Poisson distribution. Events in a time interval can be seen as random trial with a binomial distribution which is well known converges to the Poisson distribution. Therefore, it is natural to use of Poisson spike generators to generate randomized time stamps for the sensor outputs. Finally, the sensor output values will be generated using probability distributions based in *apriori* assumptions like the output range for each sensor.

An immediate benefit of synthetic data generation is the degree of control that can be exercised. For example, we can control what patterns of activity are going to be generated, what kind of noise can be added to the patterns of activity, etc. This is something that can only be done in a really limited way in a real pervasive space. For this and other reasons, we believe that the proposed idea to simulate to events or sensor outputs in smart spaces is worthwhile.

This paper is divided in the following way. In section II some basic concept to understand the idea behind the algorithm are reviewed. Section III gives a basic overview of the proposed algorithm for the generation of the data. Section IV describes in deep the proposed algorithm with the necessary description of the mathematical devices used for the data generation. In section V, we describe the events to be generated by using the new algorithm. In the following section (Section VI), we describe a possible pervasive space floor plan together with a possible collection of sensors. In section VII, we describe the basic pattern of activities to be generated. Finally in section VIII, we show some of the results of running the proposed algorithm.

II. Basic Concepts

A. Markov Chains

A Markov chain [2][11] is defined as follow:

Definition

*A **Markov chain** is a stochastic process where given the present state, future states are independent of the past states (The Markov property).*

A discrete Markov chain can bee seen as a state machine where each node represent the present state of the chain, and each directed edge has an assigned probability that represent the probability of moving from one state to the next. The collection of outward edges from a node to the other nodes can bee seen as a multinomial distribution [4]

$$f(x_1, \dots, x_k | n, p_1, \dots, p_k) = \frac{n!}{x_1! \dots x_k!} p_1^{x_1} \dots p_k^{x_k} \quad (1)$$
$$\text{with } \sum_{i=1}^{k} x_i = n$$

where each probability p_i is the probability assigned to a specific outward edge. This kind of representation allows us to naturally map pattern of activities to Markov chains. How? Because, we can see patterns of activity as a series of events such that they produce a meaningful result. This is why we decide to use Markov chains to generate the pattern of activities.

B. Spike Poisson Generator

In this section, we will describe the basic ideas behind the spike Poisson generator [7] from the point of view of homogenous Poisson process.

Given an interval $(0, T)$, we place a spike in the interval at random. Then, the probability that a spike occurred in a sub-interval (t_1, t_2) is $\frac{(t_2 - t_1) = \Delta t}{T}$. In this way, given k spikes, the probability of n of them fall in (t_1, t_2) is

$$P(n\ spikes\ in\ \Delta t) = \binom{k}{n} p^n (1-p)^{k-n}, \quad (2)$$

a Binomial distribution. Now, if we increment k and T keeping the ratio $\frac{k}{T}$ constant, it is well known that when $k \longrightarrow \infty$, equation (2) converges to the Poisson distribution

$$P(n\ spikes\ in\ \Delta t) = e^{-r\Delta t} \frac{(r\Delta t)^n}{n!}. \quad (3)$$

Then, we can simulate a succession of spikes in a random time by observing that for a single spike equation (3) becomes

$$P(1\ spike\ in\ \Delta t) \approx r\Delta t. \quad (4)$$

This allow us to simulate the succession of time spikes by subdividing the time T in m sub-intervals, and generating m random numbers $x(i)$ from the uniform distribution over [0,1]. Then:

i. If $x(i) \leq r\Delta t$ generate spike.

ii. If $x(i) > r\Delta t$ do not generate spike.

This is simply the accept-reject algorithm [11] for random sampling generation.

Given all these tools, we are ready to describe the data generation algorithm.

III. Proposed Algorithm

Here, we will describe the basics of the proposed data generation algorithm.

Synthetic Data Algorithm

---

i. Set TimeStamp = 0. This is done to have a point of reference for each element of the Markov chain.

ii. Set NODE = Initial Node. Normally, the initial node represents the beginning of a specified activity.

iii. Set N. This is the number of sensor outputs to be generated out of restricted set of possible sensor given a series of probabilities

iv. $Chain = \emptyset$. A storage for the Markov Chain

v. For i = 1 to N

   a. Sample a multinomial distribution with state NODE and probability $p_1, \dots, p_m$.

   b. Change NODE = Sample.

   c. Then:

      i. Generate a time stamp T by using the spike Poisson generator.

      ii. Generate a value O using the distribution for the node.

      iii. $Chain = Chain \cup \langle NODEID, T, O \rangle$

vi. End For

IV. Synthetic Data Algorithm

Here, we describe steep by steep each of the stages of our proposed algorithm for the generation of synthetic data.

A. Creation of Pattern Activities

In order to create meaningful pattern of activities for the sensor outputs, we use finite state machines for the creation of Markov chains [2][11]. The directed edges in the state machines have the following properties:

i. Given node i the sum of all the outward probability edges is

$$\sum_{j=1}^{n} p_{ij} = 1 \qquad (5)$$

ii. The transition probabilities between one node to the rest of the nodes are multinomial distributions,

$$f(x_1, \dots, x_k | n, p_1, \dots, p_k) = \frac{n!}{x_1! \dots x_k!} p_1^{x_1} \dots p_k^{x_k} \qquad (6)$$

We use these probabilities to represent specific behaviors. For example, for a random activity using the walking sensors, the chair sensor and the table sensor, we have the following state machine (Figure 1):

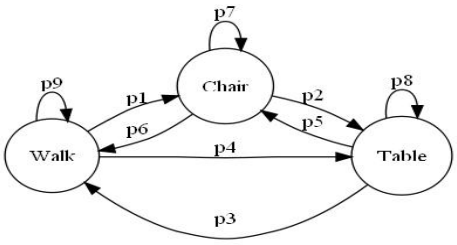

Figure 1 Example of a state machine for a random event.

In this case, for example, the probabilities

$$p6 = p2 = p7 = 1/3$$

represent equal probability to move from the spike Poisson generator Chair to the spike Poisson generators Table, Chair and Walk. Given these probabilities, it is clear that no specific behavior of activity is represented by this state machine. Another example is in (Figure 2 Example of person walking around.):

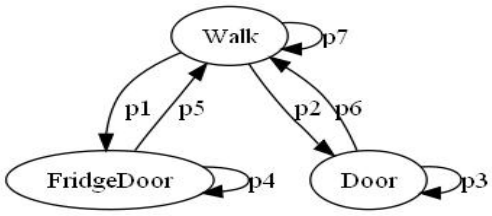

Figure 2 Example of person walking around.

In this case if we select:

$$p1 = p2 = 1/10, p3 = p4 = 1/10,$$

$$p5 = p6 = 9/10, p7 = 8/10.$$

These values in this state machine could represent a person walking around the house and opening a door and the fridge door from time to time.

In this way, we can represent specific patterns of activity.

B. Using Homogenous and Inhomogeneous Poisson Spike Generators For Time Stamp Generation

In order to generate the time stamps for the sensor output information, we will use a series of spike Poisson generator for each sensor [7]. For example, given a floor sensor, we can use an homogenous or inhomogeneous Poisson distributions for the spike generation,

$$P[n\ spikes\ in\ \Delta t] = \frac{e^{-r\Delta t}(r\Delta t)^n}{n!}, n = 0,1, \dots \qquad (7)$$

$$P[n\ spikes\ in\ \Delta t] = \frac{e^{-r(t)}(r(t))^n}{n!}, n = 0,1, \dots \qquad (8)$$

to generate the time stamps of on the sensor output.

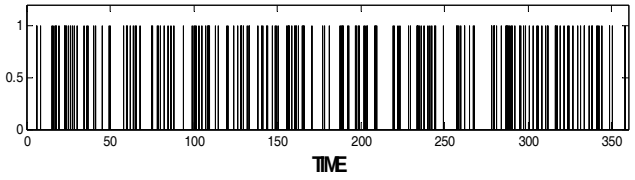

Figure 3 Each Spike represent a steep in the time domain.

Examples of the spikes generated by the homogenous Poisson process are in (Figure 3). This allows us to generate trains of timestamps for the sensor outputs in a realistic way. We still need to do more processing needs to be done to obtain a realistic simulation of the sensor output values.

However, we still need to solve one issues: (1) the generation of the sensor output at each time spike; (2) the generation of sensor outputs that correspond to the three activities to be tested. The following sections deal with these particular problems.

C. Using a Normal Distribution for the Generation of the Value Parameters

Each value parameter is generated by a Normal distribution [4] with mean $\frac{a_{g_i}+b_{g_i}}{2}$ and variance $\frac{Length(R_{g_i})}{T}$ where a_{g_i}, b_{g_i} are the lower and upper limits of the goal range R_{g_i}, and T is a convenient value for the variance. These ranges have been preselected, and they assume some knowledge about the possible sensor outputs range outputs. An example of a Poisson process of a person walking can be seen in The final values together with the times can be seen in (Figure 5).

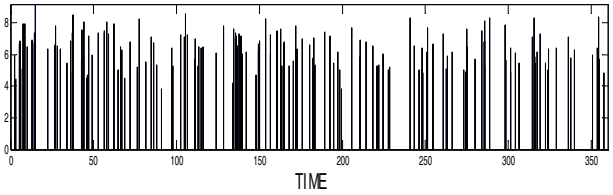

Figure 4 Time stamps together with values.

However, we still need to solve one issue, the generation of the sensor output at each time spike. For this, we propose the use the range values in a Gaussian distribution. The following section describes the method used. However, more complex distribution could be used to describe the values for the outputs of each sensor.

D. Using a Normal Distribution for the Generation of the Value Parameters

Each value parameter is generated by a Normal distribution with mean $\frac{a_{g_i}+b_{g_i}}{2}$ and variance $\frac{Length(R_{g_i})}{T}$ where a_{g_i}, b_{g_i} are the lower and upper limits of the goal range R_{g_i}, and T is a convenient value for the variance. These ranges have been preselected, and they assume some knowledge about the possible sensor outputs range outputs. An example of a Poisson process of a person walking can be seen in The final values together with the times can be seen in (Figure 5).

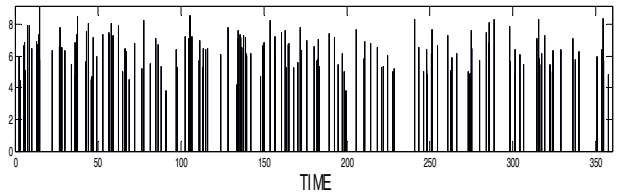

Figure 5 Time stamps together with values.

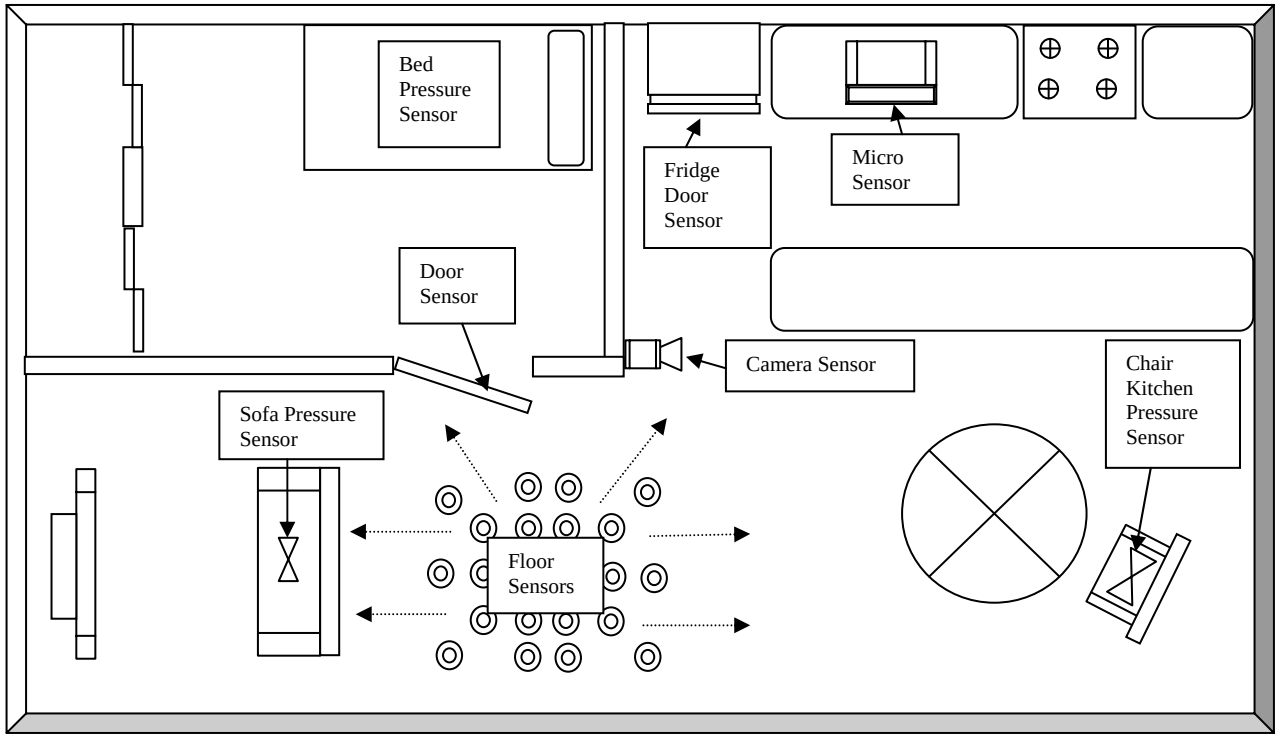

Figure 6 House floor plan.

V. Design of the Experiments

A. Events Generated Using the Model

The events to be generated follow the activities in (Table 2). For example, the *inactive* state machine may have the following structure:

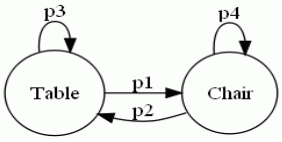

Figure 7 Example of an inactivity state machine.

In this state machine the probability transitions have the following values,

$$p1 = 0.75, p3 = 0.25, p4 = 0.80, p2 = 0.20.$$

The values *p1* and *p2* represent the high probability of the person to stay quietly in a single place. For example, this activity could represent a person sipping coffee for certain amount of time in the table and chair of the kitchen.

In addition, a series of random noise patterns are generated to test the robustness of our programmatic sensor fusion algorithm. In these random patterns all the outward probabilities have the same value. This represents the total lack of a discernible pattern in the random event.

VI. Simulated House Plan Floor for Sensor Deployment

For the sake of simplicity, we consider that the smart house model in (Figure 6) has a single occupant. More complex scenarios can be conceived, but assuming a single occupant will decrease the amount of initial constraints for the simulation of the synthetic data. Now, we will describe the collection of sensors to be used.

A. Description of a Subset of Sensor in the House Floor Plan

The following list of sensors is considered. They are extracted from the total collection of sensors in the smart house model (Table 1). They are representative of all the possible sensors that can be installed in a pervasive space. In addition, we assume that a basic interface between the hardware and the middleware is already in place converting each sensor into a singleton virtual sensor.

Table 1 Table of Sensors in floor plan

No	Sensor	Sensor Type	Range Value
1	Floor Sensors	Pressure Sensors	$[0,M_{Wal}]$
2	Fridge Door Sensor	Open/Close Sensor	[0,1]
3	Microwave Sensor	On/Off Sensor	[0,1]
4	Chair Kitchen Sensor	Pressure Sensor	$[0,M_{Ch}]$
5	Table Sensor	Pressure Sensor	$[0, M_T]$
6	Chewing	Chewing Sensor	$[0, M_{Chew}]$
7	Couch Sensor	Pressure Sensor	$[0, M_C]$
8	Bed Sensor	Pressure Sensor	$[0, M_B]$

In (Table 1), each of the values M_i represents the maximum achievable for a specific pressure sensor. These values are set by the sensor designer.

Although this is a small collection of all possible sensors used in the house floor plan (Figure 6), they are more than enough to test the feasibility of the new procedure. These sensors are used in the following activities (Table 2).

Table 2 Table of Activities

Activity	Sensors Involved					
Eating	1	2	3	4	5	6
Active	1	4	6			
Inactive	4	5	6	7		

These activities are a good way to test the feasibility of our programmatic algorithm. Now, it is necessary to discuss the generation of the time stamps and the output values for the pattern of activities.

VII. Activities to be Generated

In this section, we describe the type of activities simulated by using matrix notation for the probabilities between the edges in the state machines.

A. Eating

Using the sensors assigned to the activity in (Table 2), we define the following state machine probabilities:

Table 3 Probability description of Eating State Machine

P	Floor Sensors	Fridge	Micro	Chair Kitchen	Table	Chewing
Floor Sensors	0.6	0.2	0.2	0	0	0
Fridge	0.6	0	0.2	0.2	0	0
Micro	0.6	0	0	0.2	0.2	0
Table	0	0	0	0.2	0.7	0.1
Chair Kitchen	0	0	0	0.3	0.4	0.3
Chewing	0	0	0	0.3	0.4	0.3

This machine describes somebody walking toward the fridge, opening the fridge, putting some food in the microwave and sitting at the table. Then, the person starts to chew the food.

B. Active –Walking

In this case, we assume that a person is walking from the chair in the kitchen toward the couch. The state machine looks like

Table 4 Walking State Machine

P	Kitchen Chair	Floor sensor	Sofa
Kitchen Chair	0.3	0.7	0

Floor Sensor	0	0.4	0.6
Sofa	0	0.1	0.9

C. *Inactive – Kitchen Chair*

This inactivity event represents a person sitting at the kitchen chair sipping some coffee in the table. The machine was already described in (Section A).

D. *Inactive – Sofa*

In this case, we only have one node for the state machine. The state machine looks like,

Figure 8 Example of "*Rest in Sofa*" state machine.

In this case p1 = 1 for all the transitions.

E. *Random Noise*

Here, we simple use a fully connected directed graph to describe all the sensors, and assign equal probabilities to each of the edges such that all the outward edges.

VIII. Results of Running the Simulation

Here, we show some of the possible Markov chains using the algorithm proposed. In this section, we are

A. *Eating*

The (Figure 9) shows a Markov chain example plotted between the values and the respective time stamps. Each of the classes has a specific marker, for example the fridge door is represented by a red circle. In addition, the Markov chain is connected by a gray plot to reinforce the idea that this is a succession of events.

A total of 111 sensor outputs were generated. They show how the "eating event" is divided in two sections:

i. The first part has time from 0 to 11.43 seconds approximately. In this par the person walk between the fridge and the microwave to the kitchen table. This represents the preparation of food before the actual eating.

ii. From the 11.43 to 101.68 seconds the person is eating in the kitchen table. Here, the person is actually eating at the kitchen table. It is clear that we will receive sensor output from three main sensors: the chair, the table and the chewing detector.

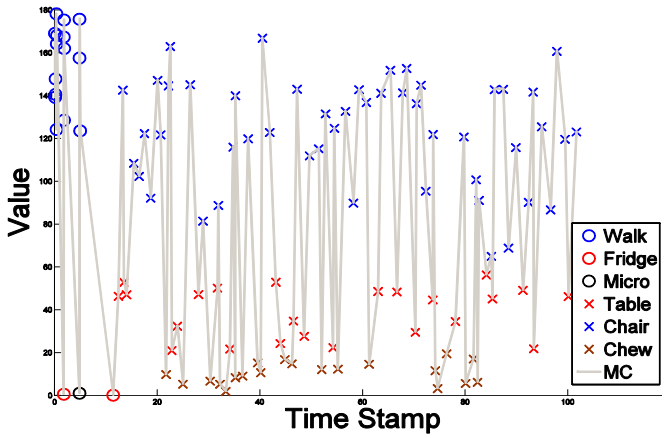

Figure 9 Example of a Markov chain for the eating activity.

B. *Active – Walking*

In this example (Figure 10), a person walks slowly around the house opening the fridge door and one the doors for around 10 minutes. The plot could represent the walking of an elder person or a person with disabilities. Other examples can be generated where a faster walking is simulated with a more homogenous walking by tweaking the values of the spike Poisson generator.

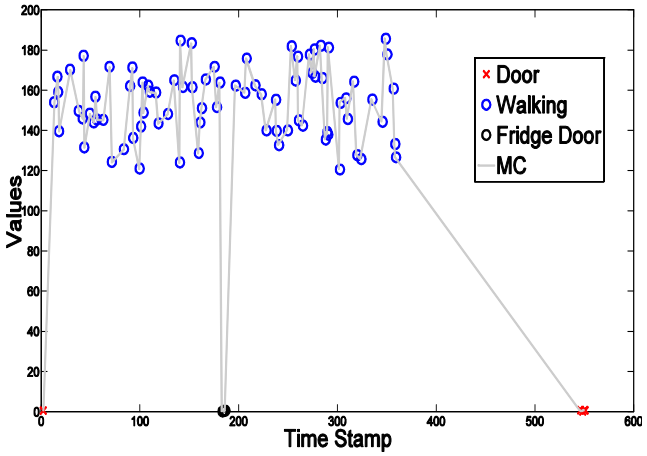

Figure 10 Example of an elder person walking through the house.

C. *Inactive – Kitchen Table*

In this section, we show a person a resting in the kitchen. As we see in the graphic the person does not trigger the chewing sensor. The person is maybe seeping some coffee or reading a book in the kitchen.

D. *Inactive – Sofa*

This is the simpler scenario (Figure 12) in this collection of examples. In this scenario a person is simply sitting and resting at the couch in front of a tv.

IX.

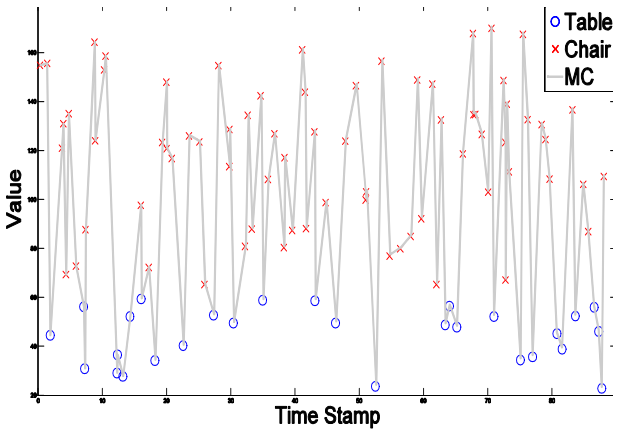

Figure 11 Example of person resting in the kitchen.

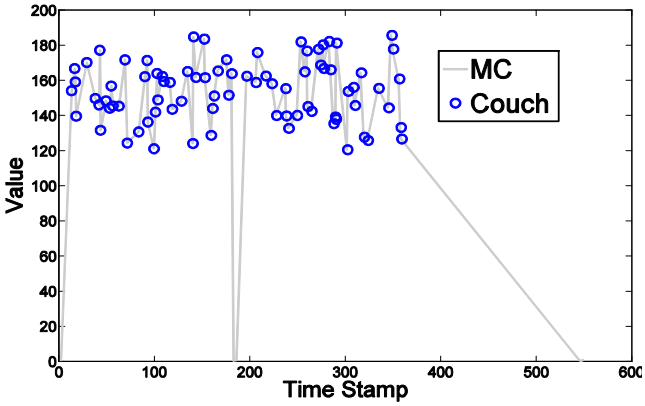

Figure 12 Example of a person resting at a couch.

X. Validation by Dynamic Time Warping

In this section, we present a real chain of observations from gathering of data using an actigraph [17] unit to collect data from a subject activity. Each entry in the data set contains four fields,

i. **Label -** sleeping, walking, reading, physical exercise, house working.

ii. **Time** – in hours.

iii. **Outdoor temperature** – in Fahrenheit.

iv. **Energy expenditure**.

It is natural to use the labels as our nodes in the state machine for the Markov chain. In this particular case we did not use the spike Poisson spike generator because the discrete times are measured in hours in the data set. The state machine for the simulation of a 50 hour period in the life of a person looks like (Figure 13).

The edge probabilities and sensor range activities were assigned to try to resemble in the most faithful way the real activity graph (Figure 14). In this activity graphic the X axis is the time axis, the Y axis corresponds to the calories consumed, and different symbols are used to label each point in the activity graphic. We call this state machine as the Proposed State Machine (PSM).

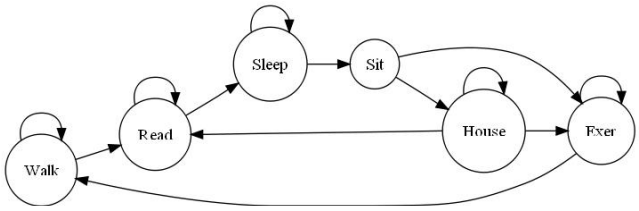

Figure 13 State machine for the simulation of activities in a 50 hour period.

We can see that the simulated activity graph (Figure 15) resembles the real activity graph in the succession of patterns of activity. For example, it is possible to see from 28 to 45 in (Figure 14) that the person from walk to read, and then finally to sleep. This is similar to the patterns of activities in (Figure 15) from 20 to 32. In order to quantify these similarities in the patterns, we use dynamic time warping [18].

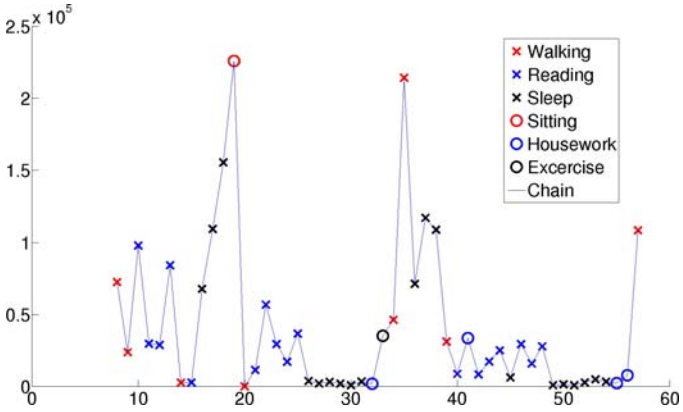

Figure 14 The activity graph to be simulated.

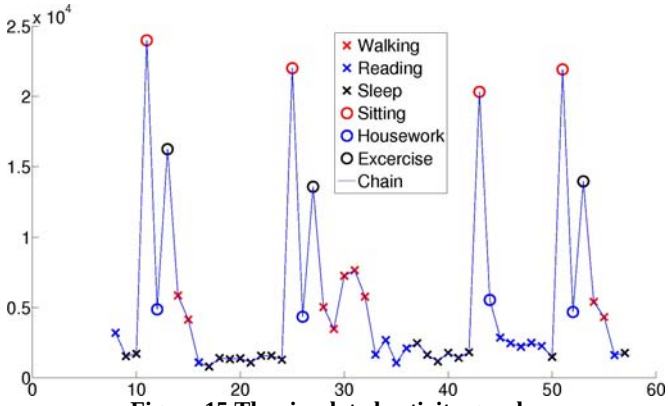

Figure 15 The simulated activity graph.

In dynamic time warping two patterns are given to the algorithm:

i. $Q = q_1 q_2 \dots q_k$.

ii. $C = c_1 c_2 \dots c_l$.

We sought to minimize the following metric cost:

$$D(Q, C) = \frac{\sqrt{\sum_{i=1}^{n} w_i}}{K}, \quad (9)$$

by using dynamic programming. The sum term $\sum_{i=1}^{n} w_i$ in the numerator represents the total number of changes made to convert one string into another, and the denominator K is the length of the warping path. (Figure 16) shows the different values for the defined metric, starting with a constant string. This string can be simulated with a state machine of one node and one edge. We will call this machine a Constant State

Machine (CSM). Then, we introduce one node and the necessary edges at a time from the proposed state machine. At each step, we simulate a string of equal size than the activity graphic to be simulated after we add the necessary node and edges, and we compare with the real activity pattern.The final results are at in (Figure 16). The CSM produces the worst value with 8.68. The PSM produces the pattern with the minimal value 0.253.

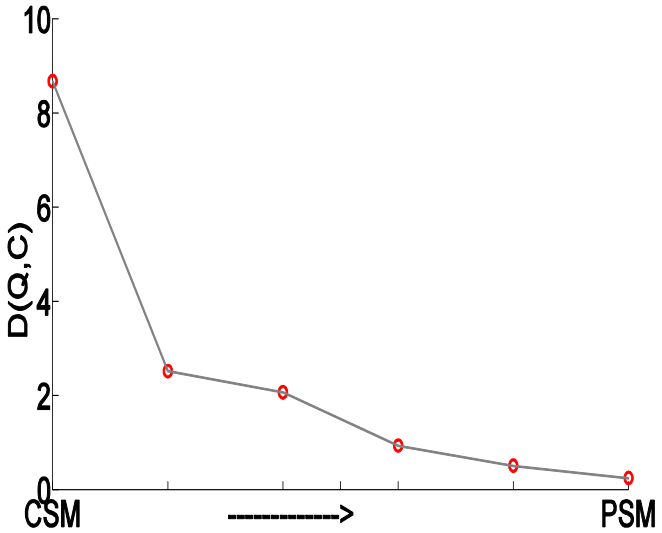

Figure 16 Values produced by the proposed metric going from the constant state machine to the proposed state machine.

Another example can be seen in (Figure 17). In this case, we start with a Random Pattern (RP). This pattern is produced using a state machine where all transition edges have the same probability. The RP produces a metric value of 1.2346 against 0.4706 in the synthetic one. An observation is that a state machine that contains half of the structure of the PSM produces the value 7.36 when compared with the original data pattern.

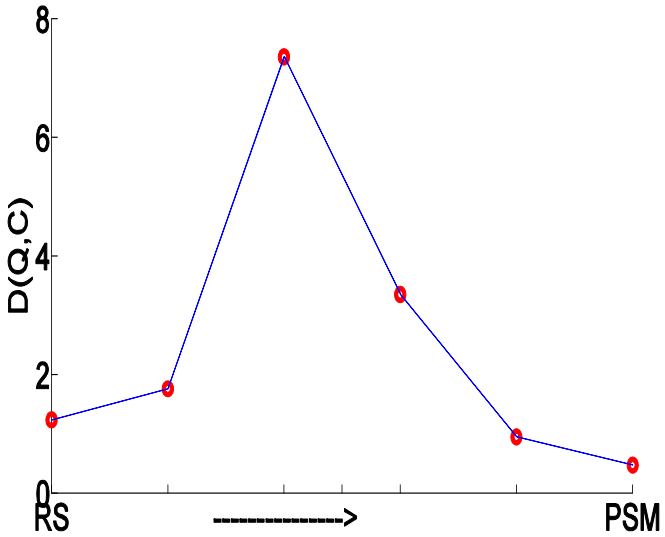

Figure 17 Values produced by proposed metric going from the random state machine to the proposed state machine.

XI. Validation by Classification Algorithms

In this section, we use the same data and state machine from the previous section to produce enough activity patterns. Then, several algorithms are run against this synthetic data, and the results are compared with real data.

Dr Cook here is your section…

XII. Conclusion

This simple technique is quite useful to generate a chain of events with particular patterns of behavior. This technique allows us to generate precise behaviors without the inherent cost of gathering data in a pervasive space. In addition, we can add precise noise to test the robustness of any new algorithm to be used in the pervasive space. However, we still need more research in the field of human behavior before we are able to have a more precise and realistic simulation of the actions of human beings under certain constraints.

XIII. References

[1] I. Armac and D. Retkowitz, "Simulation of Smart Environments," *IEEE International Conference on Pervasive Services*, pp.257-266, 15-20 July 2007.

[2] P. Bremaud, *Markov chains. Gibbs fields, Monte Carlo simulation and queues*, 3ed ed., New York:Springer, 2008.

[3] D.R. Cox and V.I. Isham, *Point Processes*, Chapman & Hall, 1980.

[4] G. Casella and R. L. Berger, *Statistical Inference*, 2nd ed. Pacific Grove: Duxbury Press, 2002.

[5] S. K. Das, N. Roy and A. Roy, "Context-aware resource management in multi-inhabitant smart homes: A framework based on Nash H-learning," Pervasive and Mobile Computing, Volume 2, Issue 4, Special Issue on PerCom 2006, November 2006, Pages 372-404, ISSN 1574-1192.

[6] A. Dupuy, J. Schwartz, Y. Yemini, and D. Bacon, "NEST: a network simulation and prototyping testbed," *Communications of ACM* 33, 10 (Oct. 1990), 63-74.

[7] D. Heeger, "Poisson Model of Spike Generation," handout, Sept. 2000. http://www.cns.nyu.edu/~david/handouts/poisson.pdf

[8] T.G. Kim, *Theory of Modeling and Simulation*, 2nd ed. Academic Press, 2000.

[9] A. Law and W. D. Kelton, *Simulation Modeling and Analysis*, McGraw-Hill, 1999.

[10] G. Mazeroff, "Markov models for application behavior analysis," in *Proceedings of the 4th Annual Workshop on Cyber Security and information intelligence Research: Developing Strategies To Meet the Cyber Security and information intelligence Challenges Ahead*, May 12 - 14, vol. 288.

[11] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[12] D.L. Snyder and M.I. Miller, *Random Point Processes in Time and Space*, Springer-Verlag. 1991.

[13] H. Thimbleby, P. Cairns, and M. Jones, “Usability analysis with Markov models,” in *ACM Transactions on Computer-Human Interaction*, vol. 8, issue 2, pp. 99-132, 2001.

[14] J. Yin, Q. Yang, D. Shen, and Z. Li, “Activity recognition via user-trace segmentation,” *ACM Transactions on Sensor Networks*, vol. 4, issue 4, Aug. 2008.

[15] G. M. Youngblood, Diane J. Cook, Lawrence B. Holder, “Managing Adaptive Versatile environments,” Pervasive and Mobile Computing, Volume 1, Issue 4, Special Issue on PerCom 2005.

[16] J.W. Yoon, D.H. Jwa, J.H. Kim, H. Park and Y.S. Moon, "Gaussian Distribution for NPC Character in Real-Life Simulation," *International Conference on Intelligent Pervasive Computing*, pp.132-135, 11-13 Oct. 2007.

[17] H. Pigot, B. Lefebvre, J.G. Meunier, B. Kerhervé, A. Mayers and S. Giroux, “The role of intelligent habitats in upholding elders in residence,” *5th international conference on Simulations in Biomedicine*, April 2003, Slovenia.

[18] L. R. Rabiner and B. Juang, *Fundamentals of speech recognition*, Prentice-Hall, Inc., 1993.