Context Recognition in Multiple Occupants Situations:  
Detecting the Number of Agents in a Smart Home Environment with Simple Sensors  

Jennifer Renoux, Marjan Alirezaie, Lars Karlsson,  
Uwe Kockemann, Federico Pecora, Amy Loutfi  
Center for Applied Autonomous Sensor Systems  
Örebro Universitet, Fakultetsgatan 1  
702 81 Örebro, Sweden

Abstract
Context-recognition and activity recognition systems in multi-user environments such as smart homes, usually assume to know the number of occupants in the environment. However, being able to count the number of users in the environment is important in order to accurately recognize the activities of (groups of) agents. For smart environments without cameras, the problem of counting the number of agents is non-trivial. This is in part due to the difficulty of using a single non-vision based sensors to discriminate between one or several persons, and thus information from several sensors must be combined in order to reason about the presence of several agents. In this paper we address the problem of counting the number of agents in a topologically known environment using simple sensors that can indicate anonymous human presence. To do so, we connect an ontology to a probabilistic model (a Hidden Markov Model) in order to estimate the number of agents in each section of the environment. We evaluate our methods on a smart home setup where a number of motion and pressure sensors are distributed in various rooms of the home.

Introduction
Context-aware systems are known as a core feature of pervasive computing whereby computers can make sense of an environment and therefore react based on their observations (Wu 2003). An important part of Context Recognition (CR) system, is to recognize the activities performed by the agents in the system (this sub-task of CR is henceforth referred to as Activity Recognition, AR). In the past years, formal context models have been suggested to deal with logic-based CR, such as the ontologies SOUPA (Standard Ontology for Ubiquitous and Pervasive Applications) (Chen et al. 2004) and CONON (CONtext OOntology) (Wang et al. 2004). Though such ontologies can in theory deal with the presence of several agents in the environment, most CR systems in the literature assume single-users scenarios when dealing with user-related data (Ko, Lee, and Lee 2007; HameurLaine et al. 2015). However, as CR systems develop, they need to consider the problem of multi-occupancy.

State of the Art

The hardware side of counting

Many person-counting sensors are already commercialized and used in various situations. Those solutions range from thermal imagers and break-beams to simple mechanical barriers (Teixeira, Dublon, and Savvides 2010). Such solutions are difficult to use in smart home environments, as the devices are expensive and usually need to be installed at each possible entrance and exit, increasing the cost of setting up the smart home. Additionally those sensors are not robust to occlusion and do not offer any way of recovering from a
undetected events.

Vision-based sensors (e.g., 2D and 3D cameras, thermocameras) are very efficient for counting as they offer an extensive view of the situation at any time and several sensors can be used to cover occlusions (Pedersen et al. 2014; Vera, Monjaraz, and Salas 2016). However, these solutions are usually applied in public spaces but are not acceptable in private spaces such as Smart Homes for obvious privacy reasons.

Pervasive sensing has received much attention during the past years due to the huge development of low-power, low-cost, miniaturized sensors and wireless communication networks. These also possess a “place it and forget it” characteristic, making them ideal for Smart Home environments. These sensors are extensively used in AR and CR systems (Gu et al. 2009; Singla, Cook, and Schmitter-Edgecombe 2010; Alerdar et al. 2013), they are rarely considered for the task of counting.

The software side of counting
A significant amount of work has been done in the past two decades to enable accurate and robust people counting using cameras. Conventional methods use techniques such as background subtraction (Shu et al. 2005; Snidaro et al. 2005), object segmentation (Rother, Kolmogorov, and Blake 2004) and human feature detection (Felzenszwalb and Huttenlocher 2003). However, as explained in the previous section, camera-based solutions are not suitable for smart home. People counting with simple non-vision based sensors has received very little attention, with most studies focusing on tracking, i.e., associating a sensor measure to a person (Hsu et al. 2010; Alerdar et al. 2013), and assuming again a known number of agents. Recently, some studies focused on counting pedestrians with binary sensors and Monte-Carlo methods (Taniguchi et al. 2014; Fujii et al. 2014) but those are once again hardly usable in homes as they make use of an important number of points such as doors, stairs and elevators, that are not present in regular homes.

The framework
In this section we present a framework for agent counting using simple non-vision based sensors and ontologies. We use a logic-based reasoning system (an Ontology) to generate automatically a probabilistic model (a Hidden Markov Model), as presented in Figure 1.

General architecture
The choice of ontologies to represent the knowledge on two main arguments: (1) Ontologies offer a very rich way to describe an environment and the knowledge we have about it, as well as easy way to instantiate different environments. (2) The reasoning capabilities of ontologies allows to infer new knowledge that can be taken into account automatically in the generated HMM, thus sparing the task of aligning manually the probabilistic model with the ontology.

The role of the CR module in this framework is simply to aggregate the data received from the sensor to create higher

level information and populate the A-Box of the ontology. In a more complex CR application, this module would also perform high-level reasoning, however this is out of the scope of this paper.

Ontology
In this section, we describe the important concepts of an ontology that would allow agent counting within our framework. It is important to note that our purpose here is not to define a full ontology but simply design requirements. One of the important elements that should be present in the T-Box is an event module through which different types of events in an environment can be represented. These event types include a Manifestation type referring to those events that are directly captured from the sensor outputs. Each instance of the class Manifestation corresponds to a change of the output of a specific sensor. The parameters of this instance are set according to the property of the object monitored by this sensor. For instance, whenever the pressure on the surface of the couch is increased, the change detection component generates a Manifestation such as m:(couch, pressure, pressed, t1, t2) without including the range of sensor data. The two last parameters t1 and t2 represent the lower and the upper bounds of the time interval during which the state of the pressure sensor stays as pressed. The value of the upper bound is initiated by the lower bound and continuously increased till the state of the sensor changes.

The Description Logic (DL) definition of the class Manifestation as used in our implementation is as follows:

\[
\text{Manifestation} \sqsubseteq \text{Event} \sqcap
\]

\[\exists \text{hasParticipant.SensingProcess} \sqcap
\]  
\[\exists \text{isEventIncludedIn.SmartObjectSituation} \]

Two other concepts essential for the person counting process are a TimeInterval concept and a Agent concept that represents respectively time distances between any two time points and the agent being involved in a process (e.g., an event). Other important concepts in the T-Box are concept that enable to define the topology of the environment as well as the equipment present, such as furniture and sensors. To do so, we defined the class Section referring to the different sections monitored, such as the bedroom, the livingroom,
Hidden Markov Models work on two important assumptions. First, the Markov assumption that states that the current state depends only on the previous state: \( P(q_t | q_{t-1}) = P(q_t | q_{t-1}) \). Second, the independence assumption, which states that the observation produced at time \( t \) is independent from previous observations and states: \( P(o_t | q_{t-1}, q_{t-1}) = P(o_t | q_t) \). Using this model, the decoding task aims to discover the most probable hidden state sequence given an observation sequence. The Viterbi algorithm (Forney 1973) is one solution commonly used for decoding. A complete theoretical overview of Hidden Markov Processes can be found in (Ephraim and Merhav 2002).

In this paper, we focus on defining the general structure of the HMM, which can be generated from information in the ontology. We assume a fixed number of rooms, denoted \( NR \), and a maximum number of persons that the system can consider, denoted \( NP \). Let \( R = \{r_1, \ldots, r_{NR+1}\} \) be the set of all \( Sections \). In order to close the environment, i.e., that the total number of persons in the environment remains constant and only the position of the persons changes, we artificially added one more room in the environment, the \( Outside \), connected to each \( Section \) with the \( AccessRoom \) property.

We define \( X = \{x_1, \ldots, x_{NR+1}\} \) as a set of state variables, each associated to one \( r_i \). Each variable \( x_t \) represents the number of persons actually present in the room \( r_i \) and its domain is \( DOM(x_t) = \{0, \ldots, NP\} \). We also define \( FOI = \{foi_1, \ldots, foi_{NF}\} \) the set of all the \( FeatureOfInterests \) defined in the ontology and \( foi_{ri} \) the set of all the \( FeatureOfInterests \) associated to \( Section \) \( r_i \). Each variable \( foi_k \) represents the value (true or false) of the \( FeatureOfInterest \), e.g., true for a pressed pressure sensor and false for a non-activated motion sensor. Therefore, \( DOM(foi_k) = \{true, false\} \).

Given this factored representation, our HMM is defined as follows:

\( S = \{s_0, \ldots, s_N\} \) is the set of states, in which \( s_i \) is one possible instantiation of the variables in \( X \), \( s_i = (s_i(r_1), \ldots, s_i(r_{NR})) \), such as \( \sum_{r_j \in R} s_i(r_j) = NP \). Each \( s_i(r_j) = x_j \) represents the number of persons present in the area \( r_j \) when the environment is in state \( s_i \).

\( V = \{v_0, \ldots, v_M\} \) is the observations alphabet. Each \( v_k \) is one possible instantiation of the variables in \( FOI \). Each \( v_k(foi_n) \) represents the truth value of the \( FeatureOfInterest \) \( foi_n \) according to observation \( v_k \). By construction, we have \( |V| = 2^{NK} \) symbols in the alphabet.

\( Q = q_1, \ldots, q_T \) is a fixed state sequence of length \( T \) and \( O = o_1, \ldots, o_T \) is a fixed observation sequence of length \( T \).

\( \pi = [\pi_i], \pi_i = P(q_1 = s_i) \) is the initial probability array. Without prior information, \( \pi \) is a uniform distribution.

\( A = [a_{ij}], a_{ij} = P(q_t = s_j | q_{t-1} = s_i) \) is the transition matrix, storing the probability of state \( s_j \) following state \( s_i \). For the sake of readability, we will often simplify the notation as \( P(s_j | s_i^{t-1}) \).

\( B = [b_j(k)], b_j(k) = P(o_k = v_k | q_t = s_i) \) is the observation matrix, storing the probability that observation \( v_k \) is produced from state \( s_i \). For the sake of readability, we will often simplify the notation as \( P(v_k | s_i) \).

In the next two sections, we will detail how the transition and the emission matrices are generated. To make the explanation easier to follow, let us consider the example presented in Figure 2. An environment is made up of 2 sections, the living room and the bedroom. An \( Outside \) room is added to the model for the need of the transition matrix, as explained in the next section.

We consider that a maximum of two persons can enter this environment. Therefore, there is 6 possible states:
sureCouch and set to at least one on the bed in the bedroom. Therefore we have two pressure sensors, one on a couch in the living room and this environment is equipped with two motion sensors and indicates at least set to one.

All other transitions are considered likely. In the previous transitions. A transition between two states \( s_i \) and \( s_j \) is considered likely if equation 4 holds:

\[
|s_i(r_m) - s_j(r_m)| = \sum_{r_n \in \text{neigh}(r_m)} |s_i(r_n) - s_j(r_n)|
\] (4)

\( \text{neigh}(r_m) \) being the set of all rooms that are topologically connected to \( r_m \).

In our example, the likely transitions from the state \( s_1 \) are \( ((s_1, s_1); (s_1, s_2); (s_1, s_3)) \) and the likely transitions from state \( s_2 \) are \( ((s_2, s_1); (s_2, s_2); (s_2, s_3); (s_2, s_4); (s_2, s_5)) \). If we consider \( p_i = 0.95 \), we get the following probabilities:

\[
\begin{align*}
    a_{11} &= 0.3, a_{12} = 0.3, a_{13} = 0.3, \\
    a_{14} &= 0.033, a_{15} = 0.033, a_{16} = 0.033 \\
    a_{21} &= 0.18, a_{22} = 0.18, a_{23} = 0.18, \\
    a_{24} &= 0.18, a_{25} = 0.18, a_{16} = 0.1
\end{align*}
\]

Generating the emission matrix. The emission matrix gives the probability to receive a specific observation knowing a specific state. In theory, there exists a dependency between each \( \text{foi}(r_j) \). Indeed, an agent interacting with the \( \text{foi} \) PressureCouch – i.e. activating the pressure sensor on the couch – is likely to also interact with the \( \text{foi} \) MotionLivingroom. A dependency between two \( \text{foi} \) in two different rooms can also be observed if the rooms are adjacent. For instance, an interacting with the \( \text{foi} \) MotionLivingroom can also interact with the \( \text{foi} \) MotionBedroom if the livingroom’s and the bedroom’s motion sensors overlap. However we will consider in this paper that all the \( \text{foi} \) are fully independent from each others. Although not realistic, this assumption simplifies the model and allows us to obtain interesting preliminary results.

Given this assumption, we can simplify the emission matrix notation as follows:

\[
B = [b_i(k)], b_i(k) = \prod_{r_m \in R, \text{foi}(r_j), \text{foi}(r_m)} P(v_k(\text{foi}(r_j)) | s_i(r_m))
\] (5)

Then we need to retrieve from the ontology the different probabilities \( P(v_k(\text{foi}(r_j)) | s_i(r_m)) \). To do so, we will use the property \( \text{IndicatesAgent} \) defined earlier and define the probability \( P(s_i(r_m) | v_k(\text{foi}(r_j))) \) for the different specializations. If the property is \( \text{IndicatesExactly} n_E \), then the probability is defined as:

\[
P(s_i(r_m) | v_k(\text{foi}(r_j))) = \begin{cases} 
    p_e & \text{if } v_k(\text{foi}(r_j)) = T \\
    p_e & \text{if } v_k(\text{foi}(r_j)) = n_E \\
    1 - p_e & \text{otherwise}
\end{cases}
\] (6)

If the property is \( \text{IndicatesAtLeast} n_L \) and \( \text{IndicatesAtMost} n_M \),
\( n_M \), the the probability is defined as:

\[
P(s_i(r_m) | v_k(foi_j)) =
\begin{cases}
p_e \frac{n_{L-M}}{n_{M-N+1}} & \text{if } v_k(foi_j) = T \\
1-p_e \frac{n_{M-L}}{n_{M-N+1}} & \text{if } v_k(foi_j) = T \\
\frac{1-p_e}{n_{L-M}} & \text{if } v_k(foi_j) = F \text{ and } s_i(r_m) = 0 \\
1-p_e & \text{otherwise}
\end{cases}
\] 

(7)

The cases where there is only a property IndicatesAtLeast – respectively IndicatesAtMost – are handled by taking \( n_M = NP \) – respectively \( n_1 = 0 \).

Then we can derive \( P(v_k(foi_j)|s_i(r_m)) \) using a Bayes’ rule and use it in equation 5.

Let’s use our previous example to illustrate the process, with \( p_e = 0.9 \). Table 1 presents the probability \( P(s_i(r_m)|v_k(foi_j)) \) for the 4 FOIs. Table 2 presents the inverted probabilities \( P(v_k(foi_j)|P(s_i(r_m))) \), computed using Bayes’ law.

Using Table 2, we can compute the emission probabilities for each state. For instance,

\[
\begin{align*}
b_1(1) &= 0.690363, b_1(2) = 0.076707, \\
b_1(3) &= 0.076707, b_1(4) = 0.008523, \\
b_1(5) &= 0.076707, b_1(6) = 0.008523, \\
b_1(7) &= 0.008523, b_1(8) = 0.000947, \\
b_1(9) &= 0.038637, b_1(10) = 0.004293, \\
b_1(11) &= 0.004293, b_1(12) = 0.000477, \\
b_1(13) &= 0.000477, b_1(14) = 0.000477, \\
b_1(15) &= 0.000053
\end{align*}
\]

Experimental setup and preliminary results

Organization of the test apartment

We used a fully functional test apartment, equipped with various sensors. The map of the apartment and the sensors equipped are presented in Figure 3.

![Figure 3: Map of the test apartment with the position of the sensors](image)

The environment consists of three rooms: the living room, the kitchen and the bedroom. The space between the entrance door and the living room is not considered part of the environment. The living room is thus considered to be the access section. In this experiment, we only used motion and pressure sensors.

Sensor data is transmitted over a wireless network consisting of several XBee nodes (https://www.digi.com/products/xbee-rf-solutions/rf-modules/xbee-zigbee) and collected by a central computer. The dataset used for our preliminary tests has been manually annotated while it was recorded. The annotations have not been post-processed and could have some seconds of delay when a subject switched rooms.

Implementation and results

In this implementation, we used \( p_l = 0.7 \) and \( p_e = 0.6 \) and a maximum of 3 persons. To measure the efficiency of our system, we used four different measures:

- The precision per room \( Prec_R \): the percentage of correct guesses regarding the number of agents in each room. With a baseline random approach, we obtained a precision of 0.36.
- The precision for the whole environment \( Prec_E \): the percentage of correct guesses regarding the number of agents in the whole smart home. The baseline random approach gives a precision of 0.28.
- The average distance per room \( Dist_R \): the average difference between the guessed number of agents for each room and the number given by the annotation. The baseline random approach gives an average distance of 0.76.
- The average distance per environment \( Dist_E \): the average difference between the guessed number of agents in the whole smart home and the number given by the annotation. The baseline random approach gives an average distance of 1.03.

In this experiment, we obtained \( Prec_R = 0.44 \), \( Prec_E = 0.24 \), \( Dist_R = 0.66 \) and \( Dist_E = 0.90 \). We first observe that our system performs better than random for all the measures except the precision per environment, even though we would have expected the precision for the whole environment to be better than the precision per room. Indeed, even if the system cannot detect correctly the agent in room, it is likely that this agent is in a adjacent room, and would still be detected as being in the environment. This assumption is not reflected in the results. By analyzing more deeply the behavior of the system, we noticed that due to the fact that the living room is considered an access room, the system tends to consider that somebody left the home when several persons are in the living room – which happens often in our experiment. Considering one more section – an entrance – with a motion sensor in it could improve the results for the precision of the whole environment.

We expect that the presented results could be improved significantly by lifting the assumption of independence between the FeatureOfInterests. Indeed we observed during our experiment that the system tends to underestimate the number of persons in a given room. This behavior is most
likely due to the above mentioned assumption. For example, if two pressure sensors on two different chairs are activated at the same time within the same room, this provides two distinct evidences for a single person in the room instead of being combined as a single evidence for two persons in the room.

Although preliminary, our results show the technical feasibility of our approach and provide a baseline for future work on more advanced models that should be capable of considering the dependence between different features of interest.

### Discussion

In this paper we presented a framework to perform agent counting using simple non vision-based sensors. This work is based on the following assumptions, ordered from the least to the most restrictive: (1) The sensors offer a good coverage of the environment and a person walking in a section can be detected. (2) The maximum number of persons in the environment cannot exceed a certain number. (3) There is no overlap in the sensor monitoring between two sections and what happens in a specific room only influence the sensors present in this room. (4) All the FeatureOfInterest are independent from each others.

Due to the very small size of the environment and of the dataset, our experiment does not allow us to conclude on the global efficiency of the system. However, it shows its technical feasibility.

In future work, we would like to relax the second and the third assumptions. A first step to relax the second assumption would be to consider a maximum number of persons \( n \) and add one more step which would represent more than \( n \). This would impact greatly the way we determine which are the likely and unlikely transitions, as well as the generation of the emission matrix. Concerning the third assumption, the system needs to use geographical concepts in the ontology. These concepts would enable to model knowledge such as The bedroom is near to the Livingroom. By using this knowledge, we can modify the way the emission matrix is generated to take spatial relations and possible overlaps into account. This will however increase the complexity of the model and might raise scalability issues.

In this paper we focused our work on Manifestations that implies the presence of an agent during their time interval, such as a pressure sensor pressed. Future work should also make sense successions of Manifestations that can also indicates a human existence even if the Manifestations themselves don’t. A classic example of this pattern is a door opening and/or closing. Even though the door being open or close does not give any indication about the fact that a human is present, the succession of Manifestations DoorOpenned-DoorClosed in a short time interval usually indicates the presence of an agent during this time interval. More complicated pattern should also be investigated.

### Acknowledgment

This work and the authors are supported by the distributed environment Ecare@Home funded by the Swedish Knowledge Foundation 2015–2019.

### References


