PLCount: A Probabilistic Fusion Algorithm for Accurately Estimating Occupancy from 3D Camera Counts

Fisayo Caleb Sangogboye  
Center for Energy Informatics  
Mærsk McKinney Møller Institute  
University of Southern Denmark  
fsan@mmmi.sdu.dk

Mikkel Baun  
Center for Energy Informatics  
Mærsk McKinney Møller Institute  
University of Southern Denmark  
mbkj@mmmi.sdu.dk

ABSTRACT
The number of occupants in a whole building, a zone or a room is an important parameter when improving the energy efficiency of a building. Using camera-based 3D or thermal sensors to detect passing of count lines are becoming more and more common. However, such sensors are not perfect and errors add up over time. In this paper we present the PLCount algorithm to accurately estimate total building, zone or room counts by fusing data from multiple count lines. The algorithm applies probabilistic reasoning together with occupancy constraints to accurately estimate the total number of occupants. We evaluate the algorithm on two data-sets from a small and a large office building. The evaluation shows a considerably lower RMSE compared to the raw counts and naive correction approach with up to 86% and 70% error reduction respectively and similarity analysis of consecutive weeks demonstrate the stability of the algorithm over time. We also demonstrate the use of the data for analyzing the energy consumption of a building. By presenting more accurate algorithms for estimating total occupancy we hope to enable buildings to better serve the actual number of occupants to improve comfort and energy efficiency of buildings.

CCS Concepts
• Mathematics of computing → Distribution functions;  
• Information systems → Data cleaning;  
• Computing methodologies → Dynamic programming for Markov decision processes;  
3D imaging;  
• Computer systems organization → Sensor networks;

Keywords
Building occupancy; occupancy sensing, count estimation; occupancy correction; camera counters; probability distribution

1. INTRODUCTION
The application of occupancy counting spans several domains including retail, building management and event facilitation. In retail, occupancy counting is used to estimate shoppers per square foot in shopping malls to determine rental charges and facilitate decisions for optimizing workforce, flow management, operational efficiency and to help increase profitability. For building energy management, occupancy counting provides the basis for estimating, understanding, gauging and optimizing energy consumption and occupancy comfort with respect to the real number of occupants. For instance, the number of people in a building zone relatively determines many important parameters for building control including the internal heat generation, $CO_2$ concentration, the amount of conditioned air to be delivered to maintain the thermal comfort and air quality of the building.

A range of sensor technologies have been studied and applied to the problem of occupancy counting. One line of work has studied reusing common building sensors for occupancy counting including $CO_2$ sensors, PIR sensors, energy metering, sensors of HVAC systems or WiFi access points [2] but often these sensors only provide counts with a high Root Mean Squared Error (RMSE). For instance Kjærgaard et al. [8] report a RMSE of 21.7 for PIR in a small office building. Beltran et al. [1] explore the idea of densely deploying lightweight thermal sensors for occupancy counting in all areas of a building. For area counting this might be an option for small buildings, however, for large buildings the cost of installation will be large. A more cost efficient solution for large office buildings is to install dedicated sensors that can count the number of occupants and their direction when passing relevant boundaries(lines) in the building. Dedicated sensors for such line counting are commercially available based on technologies such as 3D stereo vision or thermal image cameras or as research prototypes with 2D cameras, e.g., SCOPES [7]. Recently, such sensors have been declining in cost making it feasible to install them for monitoring all entrances to a building.

3D camera-based counting sensors are quite accurate in the short term. For instance, Kjærgaard et al. [8] report a RMSE of 3.3 for a three hours evaluation. However, particular detection problems associated with 3D cameras includes occlusion, pixel intensity fluctuations, and poor lighting conditions resulting in false positive and false negative counts. A major issue with the erroneous counts is that they are accumulated over an entire detection period such that detection errors are propagated until another offsetting error occurs. To visualize these challenges Figure 1 plots a week
of count data for a small office building with two entrance count lines and a large office building with nine entrance count lines. In both buildings an error offset can be observed already within the first twenty-four hours. In the small building a positive offset accumulate and in the large a negative offset. To correct such data, one can apply two constraints on the building occupancy.

- Constraint 1: The number of building occupants can not be negative.
- Constraint 2: Most buildings have periods during night time where the number of occupants go to zero.

Hutchins et al. [5] and Jussi et al. [9] have explored methods for using constraints to correct count-line data. However, both of the presented methods assume religiously that counts go to zero at night, uses extensive training data and neither of them validate their methods with ground truth data.

In this paper, we present the PLECount algorithm to accurately estimate total building or zone counts from fusing count data from multiple count lines. The algorithm applies probabilistic reasoning informed by occupancy constraints to accurately estimate the total number of occupants. The method is training free and has a reasonable running time making it easily applicable for real-world use. We make the following contributions:

- Formulate the occupancy count estimation problem with count line data.
- Propose a probabilistic algorithm to estimate total building, zone or room occupancy.
- Present an implementation of the algorithm using dynamic programming to minimize the running time of the algorithm.
- Extensive evaluation results based on 3D camera count line data from two buildings. Results for both manual ground truth analysis and similarity analysis for consecutive weeks.

- A case study of using the total counts to analyse the energy consumption of a building.

2. PROBLEM FORMULATION

The problem of occupancy counting is to estimate for any time \( t \) the cumulative count of occupants \( CC \) for any zone \( Z \), in a building \( B \). A zone \( Z \) is defined by a set of count-lines either forming boundaries within a building or boundaries between \( B \) and its environment. Prior to estimating \( CC \), the change in occupancy \( \Delta C \) is computed and it represents the difference between the forward and backward transitions for any time \( t \) and for any zone \( Z \), in a building \( B \). A special case is the zone \( Z_B \) representing the whole building. Each counting sensor defines a number of count-lines \( CL \) within the sensor’s coverage area. A count-line \( CL \) is directed and is comprised of a forward \( CL_{i,f} \) and backward \( CL_{i,b} \) stream of counts. These counts for instance \( CL_{i,f} = (e_{i,f}, \ldots, e_{i,b}) \) represents timestamped transition events \( e_i \) of a number of occupants entering or exiting over a count line. Here, \( e_{i,t} \) is equal to the cumulative sum of transitions over a specified temporal granularity for a particular \( CL_i \). Depending on the defined direction of \( CL \), the operation \((\pm)\) that is relevant for estimating the transition and cumulative counts \((\Delta C_t \text{ and } CC_t)\) at any time \( t \) in a zone \( Z \), can either be a forward and backward difference operation \((\pm)\). The information about count-line operations for each \( Z \) are defined as part of a zone model \( ZM \) where each \( Z_t \) is defined as a set of tuples \([CL_{i,\pm}, \ldots, CL_{n,\pm}]\) representing the individual \( CL_i \) in \( Z \) and the operation that should be performed with \( CL_{i,f} \) and \( CL_{i,b} \) contained in \( CL_{i,t} \), where,

\[
[CL_{i,\pm}] = \begin{cases} 
+ &: CL_{i,f} - CL_{i,b}, \text{ if } CL_i \text{ direction enters } Z_r \\
- &: CL_{i,b} - CL_{i,f}, \text{ if } CL_i \text{ direction leaves } Z_r 
\end{cases}
\]

(1)

Hence, the transition \( \Delta C_t \) and cumulative count \( CC_t \) at anytime \( t \) for any zone \( Z_t \) is given by

\[
\Delta C_{x,t} = \sum_{i=1}^{n}[CL_{i,\pm}]_t 
\]

(2)

and

\[
CC_{x,t,n} = \sum_{j=0}^{n}\Delta C_{x,t,j}
\]

(3)

A miniature example based on a case building is given in Figure 2. The building comprises 5 count-lines \( \{CL_1, \ldots, CL_5\} \) and each count-line is a labeled red line with direction arrow to signify the orientation of the individual count-line. The \( ZM \) for this building is formulated as follows:

\[
\text{ZoneA} = \{(CL_1, +), (CL_2, -)\},
\]

(4)

\[
\text{ZoneB} = \{(CL_2, +), (CL_3, +), (CL_4, -)\},
\]

(5)

\[
\text{ZoneC} = \{(CL_4, +), (CL_5, +)\},
\]

(6)

Given this model, the transition for each zone can be computed as follows:

\[
\Delta C_{x,A} = (CL_{1,f} - CL_{1,b}) + (CL_{2,b} - CL_{2,f}), 
\]

(7)

\[
\Delta C_{x,B} = (CL_{2,f} - CL_{2,b}) + (CL_{3,f} - CL_{3,b}) \\
+ (CL_{4,b} - CL_{4,f}),
\]

(8)
where \( k \) is the total number of defined zones in the building configuration.

\[
\Delta C_{2D} = (CL_{4,f} - CL_{4,b}) + (CL_{5,f} - CL_{5,b}),
\]

The whole building is also configured as a zone by utilizing all entry count-lines that are boundaries between the building and its environment

\[
\Delta C_{B} = \{(CL_{1,f} - CL_{1,b}) + (CL_{3,f} - CL_{3,b}) + (CL_{5,f} - CL_{5,b})\},
\]

(10)

Subsequently, the CC for each zone can be computed using Equation 3. Finally we define the count model CM for building B as given by:

\[
CM_{B} = \{(CC_{z1},\Delta C_{z1}), ... , (CC_{zK},\Delta C_{zK})\}
\]

(11)

where \( k \) is the total number of defined zones in the building configuration.

3. PLCount Algorithm

In this paper we propose the PLCount algorithm to correct count-line data. The goal of the algorithm is to produce an estimate that corrects all negative counts and produce an estimate that has the smallest error compared to a manually surveyed ground truth. Figure 3 gives an overview of the algorithm and the individual elements and in the following we with (X) refer to the markers on the figure. The PLCount algorithm takes as input count line data measured by the particular building instrumentation (1). The raw count line data is processed to instantiate the zone model for the particular building in focus (2). The PLCount algorithm follows a dynamic programming approach to solve the count correction problem. The basis of the algorithm for calculating a solution is a probability matrix. The first step of the PLCount algorithm is to initialize the first row of the probability matrix based on available knowledge (3). The next step is to calculate the remaining rows based on measured count line data (4). The final step is a back propagation analysis to identify the most likely solution (5). The identified solution enables the algorithm to calculate an estimate for the count of occupants for each timestep.

PLCount corrects and estimates count data over several days by dividing measured count data into daily sub-samples and perform estimation on each sub-sample. Given these sub-samples, the PLCount algorithm determines the likelihood of occupancy presence in the measured building or zone at the beginning and the end of the day. This information provides a foreground knowledge of a likely count range (initialization point) for both the beginning and end of each sub-sampled day. Two parameters for the PLCount algorithm are \( t_0 \) and \( t_n \), where \( t_0 \) and \( t_n \) represents the beginning and end of count estimation. For our experimentation \( t_0 \) and \( t_n \) are initialized to be the beginning (00:00) and end (24:00) of a sub-sampled day. However \( t_0 \) and \( t_n \) could also be selected by detecting constraint 2 such that, corrections are made from the time a building was zero till the time the building went back to zero as discussed in the discussion section.

To correct the cumulative counts for each defined zone \( Z \), in the count model \( CM_{B} \), a probability matrix \( M \) with rows \((t_0, \ldots , t_n)\) and columns \((c_0, \ldots , c_m)\) is formulated. Column \( m \) is equal to Max(CC) observed during estimation time \((t_0, \ldots , t_n)\) from the raw count line data so each \( c_j \) represents a total count of \( j \). Each element \( M_{t_i,c_j} \) of the probability matrix \( M \) is the probability of \( CC_{t_i} = c_j \), given \( \Delta C_{t_0}, \ldots, \Delta C_{t_1} \).

Hence

\[
M_{t_i,c_j} = P(CC_{t_i} = c_j | \Delta C_{t_0}, \ldots, \Delta C_{t_1})
\]

(12)

3.1 Initialize Probability Matrix

The initialization of the probability matrix depends on how \( t_0 \) is selected. There are three cases for how to initialize the probabilities. Case 0: \( t_0 \) is selected at a point in time where the building is estimated to be empty. Case 1: \( t_0 \) is selected at a point in time where we have an estimate for \( CC_{t_0} \) and Case 2: \( t_0 \) is selected at a point in time where we do not have an estimate for \( CC_{t_0} \).

Case 0:

It is relatively easy to estimate if a building is empty. In our work we calculate if there is no change in occupancy during a hour and if we consider the building empty. Figure 4 shows the distribution of hours without any changes in occupancy for the two case buildings. The small building is often empty during nighttime whereas the larger one is less likely to be empty. Given that the building is unoccupied at \( t_0 \), the probability that \( CC_{t_0} \) is zero at \( t_0 \) is set to 1 representing one hundred percent likelihood. The other columns are initialized with zero to represent that there is zero percent probability for these states. Hence

\[
(M_{t_0,c_0} = 1, M_{t_0,c_1} = 0, \ldots , M_{t_0,c_m} = 0)
\]

(13)

3.1.1 Case 1 and 2:

Given that the building is occupied at time \( t_0 \), we initialize the first row in \( M \) according to a normal distribution centered around our best guess of the number of occupants in the building. In case 1 this is the previous estimate so \( \mu = CC_{t_0-1} \). In case 2 we set the mean to the largest number of negative occupancy for any \( CC_{t_0} \) in the range \( t_0 \) to \( t_n \) calculated from the raw count line data without any corrections. Thereby each entry of the first row \( t_0 \) is calculated as:

\[
P(CC_{t_0} = c_j) = \frac{1}{\sigma \sqrt{2\pi}} e^{-(j-\mu)^2/2\sigma^2}
\]

(14)

where \( \sigma \) is calculated with regards to the possible maximum count in the building to reflect a higher uncertainty.
The underlying idea behind how we calculate the probability vector and also avoids any number overflow issues. While calculating the probability matrix \( M \) we also update a second matrix as follows for efficiently back tracking the best solutions. Therefore, we define a propagation matrix \( N \) with similar dimensions as the probability matrix \( M \).

The first row \( N_{t_0} \) in the propagation matrix \( N \) is initialized with values zero such that all element \( N_{t_0,c_j} = 0 \) with \( j = (0, \ldots, m) \). Each element \( N_{t_i,c_j} \) for \( t_i \) with \( i = (1, \ldots, n) \) is calculated as follows:

Given that a

\[
P(CC_{t_i} = c_j | \Delta C_{t_i}) \cdot M_{t_{i-1},c_k} = \max \{ \forall k \in [P(CC_{t_i} = c_j | \Delta C_{t_i}) \cdot M_{t_{i-1},c_k}] \}
\]

then, \( N_{t_i,c_j} = k \), because \( M_{t_{i-1},c_k} \) from \( (M_{t_{i-1},c_0}, \ldots, M_{t_{i-1},c_m}) \) is the prior probability that yielded the highest probability value for \( M_{t_i,c_j} \).

### 3.3 Calculate Estimates by Backtracking

Given that all elements \( M_{t_{i-1},c_j} \) and \( N_{t_{i-1},c_j} \) in the probability matrix \( M \) and propagation matrix \( N \) have been computed, the estimated probabilistic counts \( CC_{t_i} \) is derived by computing a trace from \( M_{t_n} \) - the last row in matrix \( M \) to \( M_{t_0} \) - the first row in matrix \( M \) as follows:

1. We detect if there is occupancy in the measured building or zone within a one hour horizon after \( t_n \) i.e. \( (t_n, t_n + \text{one hour}) \). If there is no occupancy in the building within this time horizon, we select element \( M_{t_n,c_0} \) and the estimated probabilistic count \( CC_{t_n} = 0 \) if not we cannot constrain occupancy to zero and we select the element \( M_{t_n,c_j} \) with the highest probability in row \( M_{t_n} \) and the estimated probabilistic count \( CC_{t_n} = c_j \).

2. The subsequent previous estimated counts e.g. \( CC_{t_{n-1}} \) is derived from the propagation value in \( N_{t_{n-1},c_j} \) such that \( CC_{t_{n-1}} = N_{t_{n-1},c_j} \).

3. Repeat step 2. until row \( N_{t_0} \) in \( N \) and output \( [CC_{t_0}, \ldots, CC_{t_n}] \).
3.4 Example of Algorithm Steps

Table 1 presents an example for zone A with precomputed ΔC, σ1 and CC values from raw data.

<table>
<thead>
<tr>
<th>Time</th>
<th>Cℓ1,1</th>
<th>Cℓ1,2</th>
<th>Cℓ2,1</th>
<th>Cℓ2,2</th>
<th>ΔC</th>
<th>σ1</th>
<th>CC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0:00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.00</td>
<td>0</td>
</tr>
<tr>
<td>2:00</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1</td>
</tr>
<tr>
<td>4:00</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1.00</td>
<td>1</td>
</tr>
<tr>
<td>6:00</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1.00</td>
<td>2</td>
</tr>
<tr>
<td>8:00</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>1.00</td>
<td>2</td>
</tr>
<tr>
<td>10:00</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>1.00</td>
<td>2</td>
</tr>
<tr>
<td>12:00</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>1.41</td>
<td>4</td>
</tr>
<tr>
<td>14:00</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1.00</td>
<td>4</td>
</tr>
<tr>
<td>16:00</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1.73</td>
<td>7</td>
</tr>
<tr>
<td>18:00</td>
<td>1</td>
<td>6</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>1.00</td>
<td>8</td>
</tr>
<tr>
<td>20:00</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>9</td>
<td>-2</td>
<td>1.41</td>
<td>6</td>
</tr>
<tr>
<td>22:00</td>
<td>21</td>
<td>2</td>
<td>3</td>
<td>27</td>
<td>-7</td>
<td>2.65</td>
<td>-1</td>
</tr>
<tr>
<td>24:00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.00</td>
<td>-1</td>
</tr>
</tbody>
</table>

Table 2: Probability matrix result after probability normalization operations on each row

<table>
<thead>
<tr>
<th>(t_i/c_j)</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0:00</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
</tr>
<tr>
<td>0:10</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
</tr>
<tr>
<td>0:20</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
</tr>
<tr>
<td>0:30</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
</tr>
<tr>
<td>0:40</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
</tr>
<tr>
<td>0:50</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
<td>0.225</td>
</tr>
</tbody>
</table>

Table 3: Propagation matrix N after computing probabilities for all elements in matrix M

This section presents the evaluation results for the PLCount algorithm. The results cover an evaluation with manual ground truth data, a similarity analysis and a runtime analysis.

To evaluate the PLCount algorithm, count data were obtained from two case buildings - a small office and a large office building. The small office building is a 2500m² building that is occupied mainly for research activities. The building houses approximately 50 researchers, technical and administrative staff and is occupied on average weekdays with a maximum average occupancy count of 70 people. Three PC2 3D stereo-vision cameras from the company Xovis are installed in the small building, covering the three entrances and exits of the building. Count-lines were defined to estimate every transition in and out of the building. The large office building is a 8000m² building, it records a maximum average of 380 occupants on normal weekdays and it facilitates several types of staff and student activities. Room types in this building comprises mainly of offices, classrooms and study areas. 17 3D stereo-vision cameras are installed in this building to cover transitions to several perimeters (zones) of the building and 9 count-lines were defined to cover the transitions through the entrances and exits of the building. All cameras used for this evaluation are manufactured by Xovis cameras and runs firmware 3.2.3 (build 4).

The installed 3D stereo-vision cameras used for this study enables for validation video playback recording of all transitions. In compliance with national regulations we obtained validation recording for a single day with strict compliance to occupant’s confidentiality and these recordings were deleted after the validation exercise. Validation recording were obtained from each camera and the transitions on all count-lines defined on each camera was validated to obtain the ground truth data. The reason for only validating a single day is that it is a very cumbersome task as it involves validating each transition through a count-line for the 24 hour period. The average time spent for validating each count-line was approximately 12 hours depending on the arrival rate on the count-line. Hence, approximately 9 days were used in total to evaluate one day of validation recording for all 16 count-lines in the two buildings.
Error in number of occupants and the Average Root Mean Squared Error (ARMSE) are used to estimate and compare the accuracy of the algorithm. We favor the use of ARMSE over RMSE because RMSE can only be interpreted with a prior knowledge of the duration of detection while ARMSE highlights the RMSE per time e.g. per minute.

In our evaluation we benchmark the PLCount algorithm to a naive correction algorithm that corrects occupancy only based on the two occupancy constraints. The naive algorithm assumes that compliance to constraint 1 is mandatory, while compliance to constraint 2 is not, given that occupancy in the building may be greater than zero at \( t_n \). Benchmarking to this algorithm indicates the advantages offered by PLCount. The naive correction algorithm corrects count data in two phases:

- **Phase 1:** The first phase corrects instances where \( \Delta C_{t_i} < 0 \) for a particular zone \( Z_i \). Given that set \( c \) is the set of all \( t_i \) where \( \Delta C_{t_i} < 0 \) such that \( c = t_{c_1}, \ldots, t_{c_k} \), each count event \( e_{t_{c_d}} \) in any count-line \( CL_{i,f} \) or \( CL_{i,b} \) in \( Z \), before any \( t_{c_d} \) is iteratively increased by 1 if \( e_{t_{c_d}} \) is from any count \( CL_{i,f} \) forward count in \( Z \), or decreased by 1 if from \( CL_{i,b} \) backward count in \( Z \). This increment or decrement is certified correct if the \( \Delta C_{t_i} > \Delta C_{t_i} \) of other zones in building \( B \) else the operation is reversed for that count event \( e_{t_{c_d}} \). After each successful operation \( \Delta C = \Delta C \). This operation is performed until all \( \Delta C_{t_i} < 0 \) for all zones in building \( B \) is resolved.

- **Phase 2:** The second phase tries to reduce the value \( \Delta C_{t_n} \): \( \Delta C_{t_n} \to 0 \) for all zone. This is achieved by decreasing each count event \( e_{t_{c_d}} \) by 1 if \( e_{t_{c_d}} \) is from any count \( CL_{i,f} \) forward count in \( Z \), or increasing by 1 if from \( CL_{i,b} \) backward count in \( Z \). Also this increment or decrement is certified correct if constraint 1 is not violated for any zone in the building and if \( \Delta C_{t_n} \to 0 \) for the building else the operation is reverted. This operation is stopped if both constraint 1 and 2 are satisfied at any time or if all count elements \( e_{t_{c_d}} \) in the building has been inspected.

### 4.1 Ground Truth Results

In this section, we discuss the error per transition rate and how PLCount compares to the naive approach for estimating and correcting occupancy counts.

To benchmark the raw camera performance, figures 5 and 6 highlights the error distribution associated with the transition rate per minute. It can be observed that the transition error increases as the number of transitions per minute increases. The results provide evidence for how we model our probabilities with a higher \( \sigma \) when a higher number of transitions is measured.

Figure 7 and 8 show the corrected building counts for the small office building using both PLCount and the naive approach, respectively. PLCount and the naive approach recorded an ARMSE score of 0.788 and 1.195, respectively, while the sensor count recorded an ARMSE of 3.308 for the entire detection period. Also in this building, the PLCount performs significantly better than the naive approach.

Figure 12 highlights the raw error distribution of the naive approach, PLCount and the raw sensor counts for the large building. It can again be noticed that PLCount achieved more error reduction on all quartiles than the naive approach.

### 4.2 Analysis of Similar Days

A method for judging the robustness of an algorithm is to visually inspect if the output shows the patterns you expect. Therefore, Figure 13 and 14 shows two weeks of PLCount results grouped by the day of the week for both the small office and the large office building. The overall patterns to look for is that the buildings have a higher occupancy on weekdays compared to weekends and that the highest level of occupancy is within the common work hours which in Denmark is between 8.00 and 16.00. In both buildings we can observe these patterns providing evidence for the robustness of the algorithm. In the small office building the
Figure 7: Corrected count data for small building using PLCount 25th of May, 2016

Figure 10: Corrected count data for large building using PLCount 25th of May, 2016

Figure 8: Corrected count data for small building using Naive approach for 25th of May, 2016

Figure 11: Corrected count data for large building using Naive approach 25th of May, 2016

Figure 9: Error for both PLCount and Naive correction approach for small building

Figure 12: Error for PLCount and Naive correction approach for large building
occupancy patterns are more similar among the two weeks compared to the larger building. The reason is the difference in usage of the two buildings with the small building mainly containing offices and the large building mainly containing teaching and student study zones which have a more fluctuating usage. In the large building a special event took place on Friday the 23rd week which can be noticed by the peaking occupancy.

### 4.3 Runtime Analysis

To be widely applicable an algorithm for correcting counts needs to compute corrections fast. The PLCount algorithm performs count correction in polynomial runtime of $O(n \times m)$ where $m$ is the maximum occupancy. This could be slow if the value of $m$ grows significantly. One way to optimize PLCount is to compute conditional probabilities for a possible range $R$ i.e. $P(CC_i = R|\Delta C_i)$ where $R = \{CC_{i-1} - max(||\Delta C||), \ldots, CC_{i-1} + max(||\Delta C||)\}$ and $len(R) << m$ and utilizing Piecewise Cubic Hermite Interpolating Polynomial (PCHIP) to extrapolate for the remaining uncomputed probabilities. Where $k = len(R)$, the speed improved PLCount algorithm performs count correction in polynomial runtime of $O(n \times k) << O(n \times m)$.

Figure 15 shows the runtime for both variants of PLCount as the value of $m$ increases. The values of $m$ includes 32, 85, 206, 419, 742 and the runtime for each $m$ was obtained on an Hp EliteBook 850 computer with a 2.6 GHz Intel Core i7 processor and 16 GB RAM. The speed improved PLCount utilizes averagely 10% of the runtime of the primitive PLCount with no tradeoffs on correction results.

### 5. DISCUSSION

In this section we discuss how to detect low and empty occupancy and how the calculated occupancy estimates provided by PLCount can be applied for analyzing the energy consumption of buildings.

#### 5.1 Detecting empty or low occupancy

Accurately detecting empty or low occupancy in a building could facilitate a robust back-propagation analysis for PLCount and in this paper, we investigate the use of deployed PIR sensors. Figure 16 shows the corrected counts and PIR sensor triggered for the large office building for four days and the PIR triggers indicate the number of rooms that
are occupied per time step. The results indicates that the building was completely empty only once at night and there after rises. This information can facilitate a more accurate case selection for initializing the probability matrix and also an initialization point or range for back-propagation analysis. Using PIR sensors as a zero or low occupancy indicator is even more relevant for bounding the back-propagation analysis especially during periods (primarily at nights) when low occupancy is expected. During such period, the initialization range for the back-propagation analysis could be pegged between the number of rooms occupied and the estimated count by the camera sensor. In our future work, we will experiment with such extensions of PLCount.

5.2 Occupancy versus Energy Consumption

In our paper we have motivated that having accurate counts data should be obtained. A commonly used sensor for estimating the occupancy and controlling ventilation systems is a CO₂ sensor. An obvious challenge for utilizing CO₂ sensors is the long response time resulting in detection delays. Fisk et. al. in [4] analyse the accuracy of CO₂ based occupancy counts using 44 CO₂ sensors deployed in nine commercial buildings. The study concludes that the sensors are very prone to failures and calibration errors. Another passive (implicit) occupancy detection method are introduced by Christensen et. al. [2] and Ruiz et. al. [10]. Christensen et. al. identified a partial correlation between WiFi enabled devices connected to existing network infrastructure and electricity consumption in commercial buildings and they concluded that connected devices can be a good metric for estimating occupancy in a building. Ruiz et. al. extracted spatial-temporal features from WiFi measurements to determine occupancy density and flow and to classify behavioral roles within an hospital building. One challenge with this method is that occupants are required to carry dedicated devices (mobile phones and PCs) and be connected to an access point. This resulted in a case cited in [2] where only 40% of the ground truth occupancy count was accounted for by detected host devices. Similarly challenges could be noticed in [3] which utilizes bluetooth beacons for occupancy detection in buildings. The system proposed in [3] requires active connection to installed bluetooth beacons and all gathered location data from monitored devices are sent to a server for additional processing. The accuracy recorded was only based on connected devices not on ground truth occupancy count of people in the detected location.

Kjærgaard et. al. [8], compared count data obtained from PIR sensors in a building with ground truth data. This comparison yielded a root mean squared error RMSE of 21.7. The PIR sensor recorded such low accuracy because they are a simple way of detecting if a room is occupied, thus at some point in time the number of PIR triggers was no longer sufficient to estimate occupancy count.

Thermal cameras unlike 3D counting cameras utilizes thermal imaging system to differentiates the hotter surface of occupant’s body compared to their background. While this detection method are independent of lighting conditions, regions with relatively high temperature close to the standard body temperature may yield false negative and positive

6. RELATED WORKS

To enable the opportunities afforded by count information, accurate counts data should be obtained. A commonly used sensor for estimating the occupancy and controlling ventilation systems is a CO₂ sensor. An obvious challenge for utilizing CO₂ sensors is the long response time resulting in detection delays. Fisk et. al. in [4] analyse the accuracy of CO₂ based occupancy counts using 44 CO₂ sensors deployed in nine commercial buildings. The study concludes that the sensors are very prone to failures and calibration errors. Another passive (implicit) occupancy detection method are introduced by Christensen et. al. [2] and Ruiz et. al. [10]. Christensen et. al. identified a partial correlation between WiFi enabled devices connected to existing network infrastructure and electricity consumption in commercial buildings and they concluded that connected devices can be a good metric for estimating occupancy in a building. Ruiz et. al. extracted spatial-temporal features from WiFi measurements to determine occupancy density and flow and to classify behavioral roles within an hospital building. One challenge with this method is that occupants are required to carry dedicated devices (mobile phones and PCs) and be connected to an access point. This resulted in a case cited in [2] where only 40% of the ground truth occupancy count was accounted for by detected host devices. Similarly challenges could be noticed in [3] which utilizes bluetooth beacons for occupancy detection in buildings. The system proposed in [3] requires active connection to installed bluetooth beacons and all gathered location data from monitored devices are sent to a server for additional processing. The accuracy recorded was only based on connected devices not on ground truth occupancy count of people in the detected location.

Kjærgaard et. al. [8], compared count data obtained from PIR sensors in a building with ground truth data. This comparison yielded a root mean squared error RMSE of 21.7. The PIR sensor recorded such low accuracy because they are a simple way of detecting if a room is occupied, thus at some point in time the number of PIR triggers was no longer sufficient to estimate occupancy count.

Thermal cameras unlike 3D counting cameras utilizes thermal imaging system to differentiates the hotter surface of occupant’s body compared to their background. While this detection method are independent of lighting conditions, regions with relatively high temperature close to the standard body temperature may yield false negative and positive
counts. This was evident in the estimation in [9] which utilized several deployed cameras to estimate occupancy count for DCV real-time control. A similar cumulative counting error shown in Figure 1 was observed and they were compensated for by applying two constraints namely setting negative counts to zero and resetting occupancy counts at midnight. These constraints are based on the similar assumptions made in [5] which considers negative counts as random errors and assumes that buildings are unoccupied at midnight which is similar to the behavior of the naive approach that we have shown that PLCount is superior to.

Apart from constraints applied by both [9] and [5], Ihler et al. in [6] modeled count data at a single sensor by formulating a probabilistic model for each sensor derived from an inhomogeneous Poisson process representing usual human activity. The approach also differentiates between usual activity and unusual behavior. A hidden Markov process was used to model and represent bursts of such unusual behavior. [5] extended this method to multi-sensor environment by linking individual sensor streams to form a multiple-sensor probabilistic model for building occupancy through directed graphical model. [5] trained their model with 6 weeks of count data from a campus building and validated the robustness of their model by replacing observed measurements with missing labels and with corrupted data but not with ground truth data. Conversely, our model requires no training whatsoever and we have evaluated its performance with ground truth data from two office buildings with varying sizes and usage purpose. Also coupled with the fact that PLCount does not require any training whatsoever, it has low computation time and does not strictly encode constraints.

7. CONCLUSIONS

In this paper, we presented PLCount - a probabilistic count correction and estimation approach based on dynamic programming for correcting erroneous count data. We presented the motivation for proposing the algorithm, the inefficiency of the naive approach to count correction, and various strategies of our probabilistic approach for handling special count estimation cases. We have implemented PLCount and evaluated its performance based on obtained ground truth data from two case buildings and we have benchmarked the performance of PLCount with the naive based correction approach. The results highlight that PLCount outperforms the naive based approach for both the small and large case building by a ratio of 70% and 35% respectively. Also occupancy patterns for similar days were investigated and PLCount robustly identified peculiar occupancy patterns and occupancy dynamics associated with each day, providing a flexible correction that can adjust to varying and dissimilar days. Finally we illustrated the advantages of using occupancy data for analyzing energy consumption of buildings.

8. ACKNOWLEDGMENTS

This work is supported by the Innovation Fund Denmark for the project COORDICY (4106-00003B).

9. REFERENCES