CeilingSee: Device-Free Occupancy Inference through Lighting Infrastructure Based LED Sensing

Yanbing Yang[1] Jie Hao[2] Jun Luo[1] Sinno Jialin Pan[1]
School of Computer Science and Engineering, Nanyang Technological University, Singapore[1]
College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China[2]
Email: {yyang017,junluo,sinnopan}@ntu.edu.sg[1] haojie@nuaa.edu.cn[2]

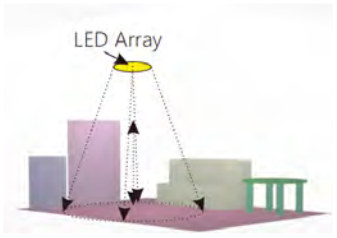
***Abstract*—As a key component of building management and security, occupancy inference through smart sensing has attracted a lot of research attentions for nearly two decades. Nevertheless, existing solutions mostly rely on either pre-deployed infrastructures or user device participation, thus hampering their wide adoption. This paper presents CeilingSee, a dedicated occupancy inference system free of heavy infrastructure deployments and user involvements. Building upon existing LED lighting systems, CeilingSee converts part of the ceiling-mounted LED luminaires to act as sensors, sensing the variances in diffuse reflection caused by occupants. In realizing CeilingSee, we first re-design the LED driver to leverage LED's photoelectric effect so as to transform a light emitter to a light sensor. In order to produce accurate occupancy inference, we then engineer efficient learning algorithms to fuse sensing information gathered by multiple LED luminaires. We build a testbed covering a 30m^2 office area; extensive experiments show that CeilingSee is able to achieve very high accuracy in occupancy inference.**

I. Introduction

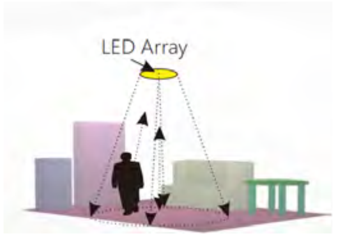
The awareness of (indoor) occupancy is crucial to many aspects of smart building management; these include, among others, controlling the HVAC (Heating, Ventilation, and Air Conditioning) and lighting systems for the sake of energy conservation, choosing the right information service based on congestion level, as well as safely evacuating people under life-threatening circumstances. In the past two decades, various smart sensing technologies have been dedicated[1] to infer occupancy for indoor facilities, and most of them require deploying certain sensing infrastructure with mainly three typical sensors: passive infra-red (PIR) [5], [6], [7], [8], acoustic/ultrasonic [9], [10], [11], and camera [12], [13], [14], [8]. Other solutions attempt to infer occupancy indirectly by monitoring the usage of existing services (e.g., Wi-Fi [15] and power grid [16]). Whereas the former method requires installations of extra infrastructure and hence incurs both high cost for building management and potential infringement of user privacy, the latter approach can hardly be accurate: what if some occupants simply do not use any services?

Our key observation is that, in any human occupied indoor spaces, lighting is a necessity while the resulting diffuse reflection can be "perturbed" by the presence of occupants. At the meantime, *Visible Light Sensing* (VLS), as a variance of heavily studied *Visible Light Communication* (VLC) [18], [19], [20], [21], has started to show its potential in many sensing-intensive applications [2], [22]. Therefore, a natural question is: *can we apply VLS to build an occupancy inference system that is free of reliance on both heavy infrastructures and user involvements?* In this paper, we intend to provide readers with a positive answer through our CeilingSee system.

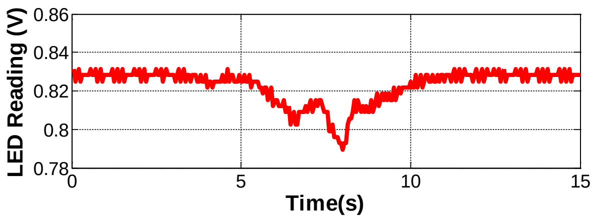
[1]Though occupancy information can be derived from an indoor localization system (e.g., [1], [2], [3], [4]), few practical indoor localization systems have been widely adopted so far. Moreover, relying on user location tracking to "count" occupancy is highly inefficient and may infringe privacy.



(a) Sensing coverage of LED array.



(b) Occupant under the coverage.



(c) Variances in LED readings due to absence/presence of occupant.

Fig. 1. Inferring occupancy by LED sensing.

The first idea of CeilingSee is very intuitive: from the ceiling point of view, the diffuse reflection (consisting of mainly reflections from the floor and various fixed furnitures on the floor) is bounded to be affected by the presence of occupants. Therefore, sensing such perturbations could allow us to infer occupancy. While simply installing an array of light sensors on the ceiling could be a solution, it would introduce yet another infrastructure. Fortunately, the increasing popularity of LED lighting systems and the readily verifiable photoelectric effect of LED [23], [19] have motivated our novel idea: re-designing the driver of a *Commercial Off-The-Shelf* (COTS) LED could enable it to serve as both a light emitter and a light sensor. Consequently, CeilingSee simply leverages the existing LED lighting systems and borrows a fraction of LED chips to sense the variance in diffuse reflection. We illustrate these ideas in Fig. 1; it clearly demonstrates the potential and effectiveness of VLS-based occupancy inference.

TABLE I
COMPARING CEILINGSEE WITH EXISTING SOLUTIONS FOR OCCUPANCY INFERENCE.

Sensor Type	Infrastructure reuse	Privacy Concern	Accuracy	Cost	Limitation
Ultrasonic [11]	No	No	$\geq 90\%$	Medium	Need prior maximum and distribution of occupants
Thermal Sensor & PIR [7]	No	No	88.5%	Medium	Extra sensors
Camera [13]	Possible	Yes	80%	High	Heavy computation, re-calibration based on dimming
Camera & PIR [8]	Partial	Yes	94%	High	Heavy computation, re-calibration based on dimming
Photosensors [17]	No	No	N/A	Low	Extra sensor, re-calibration based on dimming
LED (CeilingSee)	Yes	No	$\geq 90\%$	Low	Re-calibration based on dimming

The seemingly straightforward ideas of CelingSee impose on us two major challenges. Firstly, although conventional LED-to-LED communication [19] has already employed an LED as receiver (a special form of sensor), sensing the variance in diffuse reflection is much more challenging due to the very low SNR, hence it necessitates using the collective sensing ability of multiple LED chips. Existing LED receiver directly connects an LED chip to the I/O port of an MCU, thus relying on the controllable nature of the I/O port to toggle the states of the LED between *forward biased* (emitting or sending) and *reverse biased* (sensing or receiving). Unfortunately, this would not work when multiple LED chips are used together, as the voltage/current would exceed what an I/O can take (normally no more than 3.3V/20mA). As a result, we design a novel circuit for accommodating the collective photoelectric effects of an LED array.

Secondly, as the sensing coverage (a.k.a., *Field of View*, or FoV)) of a single LED array (consisting of multiple "sensorized" LED chips) is limited, we have to use multiple arrays to cover a large indoor area, which happens to be in line with the lighting requirement. Moreover, CeilingSee needs to account for multiple occupants dispersed on the area, especially those not strictly under of an LED array. Therefore, it is necessary that efficient inference algorithms are in place to utilize the collective sensing outcomes of all LED arrays. CeilingSee responds to this challenge by engineering a machine learning algorithm that maps the multi-dimensional sensing data to the demanded occupancy count.

To validate our design of CeilingSee, we build a testbed consisting of multiple LED arrays in order to cover a 30m^2 office area. We implement the hardware part for controlling the LED arrays, as well as the software part for sensing data processing and hence occupancy inference. Our main contributions are as follows:

- We propose the novel idea of applying ceiling-mounted LED lighting systems for inferring occupancy, and we build CeilingSee to showcase the efficiency and effectiveness of this lightweight occupancy inference approach.
- We re-design the driver of an COTS LED so that CeilingSee can freely toggle an LED between light emitting and light sensing modes and a group of LEDs can be collectively used for sensing the variance in (indoor) diffuse reflection.
- We engineer data processing and machine learning algorithms to deduce the occupancy within the FoV of an LED array, and to infer full area occupancy by fusing the multi-dimensional sensing outcomes from all arrays.
- We conduct extensive field experiments in the past six months to validate the effectiveness of CeilingSee, and the results strongly demonstrate its high accuracy in occupancy inference.

We are not expecting CeilingSee to fully replace the existing occupancy inference solutions. Instead, we deem the technical implication of CeilingSee as twofold: i) it is an avatar of a VLS-based idea that minizes the resource required for inferring occupancy and may thus inspire other applications of VLS, and ii) it serves as a complement to other solutions for improving the efficiency and scalability of occupancy inference systems. We comparing CeilingSee with typical existing solutions in Table I. As CeilingSee and an existing solution adopting photosensors [17] are both of low cost, we further compare their costs in Table II. In the following, we first introduce the principle of LED sensing, along with the design and experience with a single sensing unit of CeilingSee in Sec. II. Then we present our learning-based occupant inference algorithms in Sec. III. We report the performance evaluation of CeilingSee in Sec. IV and conclude the paper in Sec. V.

II. SENSING REFLECTION BY LED

Lighting systems are pervasively deployed and used for indoor spaces due to the inadequate natural lighting from windows, especially when the space has limited access to day light. Such lighting systems are normally ceiling-mounted and hence cause diffuse reflections from the floor (including various furniture on it), which is biased towards the ceiling due to the blending with minor specular reflection. When (human) occupants move into this "reflection field", they can cause perturbations readily sensible by certain ceiling-mounted light sensors. To avoid introducing an extra sensing infrastructure, our idea is to re-use part of the existing LED lighting system to serve the sensing function.

It is well known that LED has photoelectric effect, i.e., light shining upon an LED can cause it to emit electrons, which is a reverse effect of LED's default functionality [24].

TABLE II
COST COMPARISON OF CEILINGSEE WITH PHOTOSENSOR SOLUTION [17].

CeilingSee		Photosensor[17]	
Components	Cost	Components	Cost
Switches	\$2.76	Photosensors (TEPT5600)	\$0.7
Amplifiers	\$0.37	Amplifiers	\$3.28
MCU (CC2541)	\$4.34	MCU (MSP432)	\$7.89
Other passives	\$4.04	Other passives	$\geq$\$5.4
Total	\$11.51	Total	$\geq$\$17.27

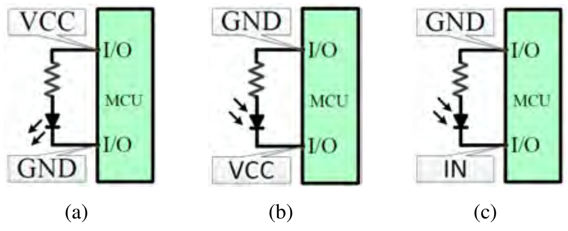


Fig. 2. Conventional bidirectional interface between LED and MCU. (a) Normal I/O configuration for light emitting. (b) Reverse bias for light sensing. (c) I/O as input for reading sensed signal.

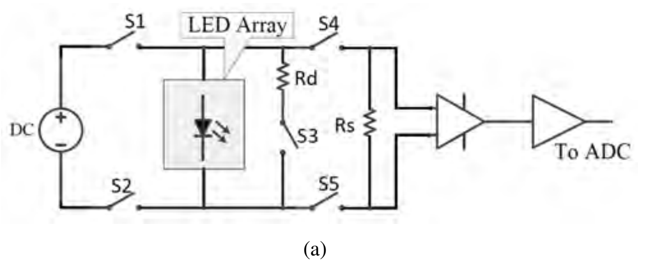
This effect has motivated a few proposals to use an LED not only as a sender but also as a receiver in VLC [23], [19]. However, converting an LED receiver to an LED sensor is almost impossible as the light signal sent by an LED has a much higher *Signal-to-Noise-Ratio* (SNR) than the variance in reflection. Moreover, as we only "borrow" part of the LED lighting system for the sensing purpose, we want to toggle the LEDs between light emitting and light sensing modes when either is in need. Therefore, we present the details of CeilingSee's hardware implementation in this section, aiming to address the aforementioned issues.

A. *From LED Receiving to LED Sensing*

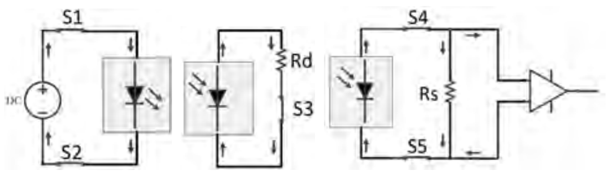
We briefly describe the conventional design of an LED receiver, and then we explain why and how the design of LED sensing for CeilingSee should be different.

1) Bidirectional Setting of LED Receiver: In a conventional design for light receiving in the LED-LED communication, a bidirectional interface to an LED is created by connecting the LED directly between the two I/O pins of a micro-controller (MCU) [23], as shown in Fig. 2. Fig. 2(a) shows that the LED emits lighting when its anode and cathode are connected to VCC and GND, respectively, via a simple I/O configuration. Reverting the I/O configuration sets the LED in reverse bias mode as in Fig. 2(b); it charges the inner stray capacitance of the LED and prepares the LED for light sensing. Fig. 2(c) further illustrates the actual measurement phase: MCU reads the voltage changes on LED's cathode and times how long it takes for the photocurrent to discharge the capacitance to the I/O pin's digital input threshold, as the discharging time is inversely proportional to the amount of incident light. The simplicity of this LED receiving circuit stems from the matching voltages between an MCU and an LED, which does not work if one wishes to drive more LEDs with one MCU.

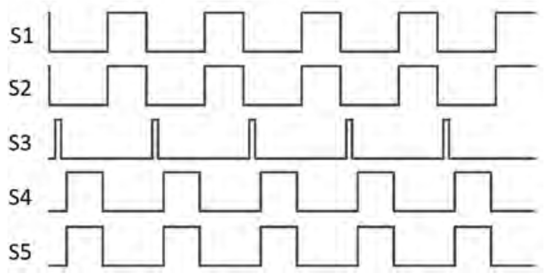
2) Collective Sensing with Multiple LEDs: Compared with the LED-LED communication, the signal for our reflection sensing scenario is much weaker.[2] As a result, if we used the same circuit as described in Section II-A1, we would either fail to sense the signal due to the too weak photocurrent or experience a huge sensing delay thanks to the long discharging time. Whereas using multiple LEDs to collectively sense the



(a)



(b) (c) (d)



(e)

Fig. 3. Architecture of the LED array driver. (a) Circuit schematic. (b) The equivalent circuit during light emitting. (c) The equivalent circuit for discharging. (d) The equivalent circuit during light sensing. (d) Duty-cycled control signal sequence for toggling between lighting and sensing.

weak signal appears to be a straightforward solution, the much higher voltage level caused by aggregating multiple LEDs renders the existing I/O-based circuit design invalid. Also, reverse biasing an array of LED is almost impossible for a normal MCU with limited absolute maximum voltage.[3]

Therefore, our intention is to use an LED array as one light sensing unit without the need for reverse biasing it. According to the LED's equivalent circuit model [23], an LED can be deemed as a current source with a shunt capacitor, while the current source is driven by incident light to generate tiny photocurrent. Whereas the current produced by one LED is too weak to be measurable (even through an amplifier), the aggregated current of the LED array (with a sufficient amount of LEDs) would suffice for the sensing purpose.

Based on the aforementioned ideas, we re-design the driver of COTS LED luminaires to control the LED state toggling, so that CeilingSee can duty-cycle part of the luminaire between light emitting (for normal lighting) and light sensing (for occupancy inference). Fig. 3 shows the architecture of this driver circuit and also its functionalities under different modes. The five switches in Fig. 3(a) are key components for the driver. The LED array emits light (lighting phase) if both S1 and S2 are ON and other switches are OFF, as shown by

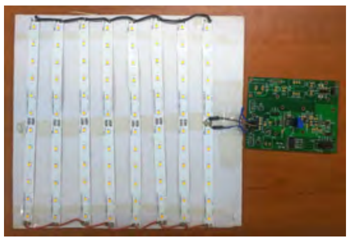
[2]The signal could also be weaker in LED-LED communication if the transmission distance goes beyond the centimeter testing scenarios in [23], [19]. As a practical sensing system, CeilingSee has to work under a "transmission distance" (that from floor to ceiling) of several meters.

[3]Though this could be made possible by using special high-power components, e.g. high reverse voltage MOSFET, the high cost would compromise our purpose of building a lightweight occupancy inference system.

Fig. 3(b). Subsequently, Fig. 3(c) shows a short discharging window that allows residual charges on the array to be cleared for preparing sensing (discharging phase), by putting S3 ON while others OFF. Finally, switching S4 and S5 ON and others OFF enables the array to act as a light sensor (sensing phase): the resistor Rs ($> 10\text{M}\Omega$) converts weak photocurrents to voltage signals that drive amplifiers to produce sensing outcome. We also show the duty-cycled control sequence of different switches in Fig. 3(e): to prevent short circuit, a dead-time should be inserted between two consecutive phases.

B. Experiencing Single Sensing Unit

Fig. 4 shows one *sensing unit* of CeilingSee; it consists of an 8×12 LED chip array [25] and a PCB carrying our re-designed driver. We separate the chips into two groups and duty-cycle them in a complementary manner to perform both lighting and sensing simultaneously while avoiding flickering. Currently, we toggle them at a long switching period (every 10 minutes) between lighting and sensing. Using this hardware platform, we first test the capability of single-unit occupancy inference, aiming to study the performance of such a unit under various parameter settings and situations. More hardware implementation details on the unit will be given in Section IV-A.


(a) Sensing/lighting uint

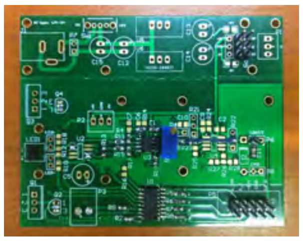

(b) Re-designed driver

Fig. 4. A sensing/lighting unit of CeilingSee, consisting of a LED array and a re-designed driver.

1) Raw Reading and Signal Smoothing: We first demonstrate the effectiveness of using LED sensing to indicate occupancy: we record V_{ADC} when one occupant walks into the sensing coverage and stay there, and the V_{ADC} in Fig. 5 clearly shows the variance in reflection incurred by an occupant. The rather unstable raw readings are mainly interfered by two sources: a minor random noise and a 50Hz component.[4] Both interferences are very easy to be removed by applying a moving average with a window size of 100ms. Therefore, we report only the smoothed readings hereafter. Also, we take the absolute difference between a V_{ADC} reading and the nominal reading (obtained in absence of any occupants, when we train or re-train CeilingSee) as the actual **sensing value**.

2) Impacts of Illuminance and Dimension: As a sensing unit is sensing the reflection, the outcome ought to be affected by the ambient illuminance. Also, according to Section II-A2,

[4]As we deploy CeilingSee in a public research lab, the LED sensing units have to be co-located with existing fluorescent lights that cause the 50Hz component. This happens to attest the compatibility of CeilingSee with legacy lighting systems.

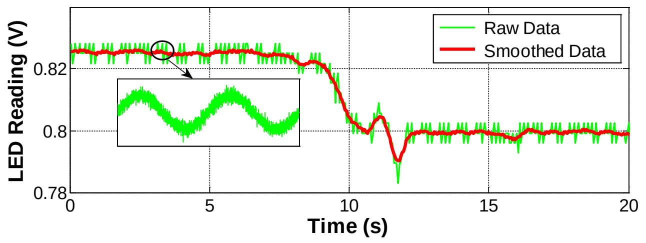
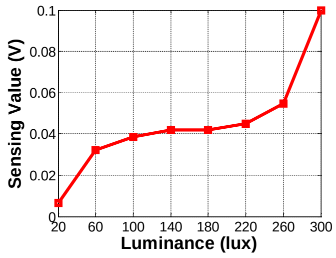
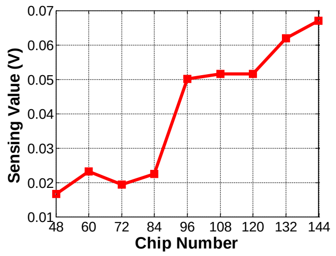


Fig. 5. Raw and smoothed ADC readings (V_{ADC}) that indicate reflection variance due to occupancy.

the sensitivity of the unit depends on the number of LED chips involved. We are here to quantitatively understand these impacts. Fig. 6(a) shows the sensing values are significantly affected by (average) illuminance, and an illuminance lower than 60lux would affect the performance of CeilingSee. Fortunately, normal office area has a light level of 200lux and other public facilities such as supermarkets or theaters can reach 1000lux [26]. In the following, we maintain the average illuminance to 150lux: the one offered by the laboratory where CeilingSee is deployed. In Fig. 6(b), we vary the number of chips and check the resulting sensitivity. The results show that the unit with 96 chips appears to be the most cost-effective choice, hence leading to our design in Fig. 4.



(a) Sensing value vs. illuminance

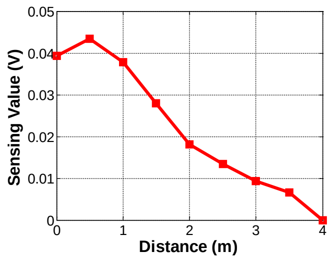


(b) Sensing vs. LED chips number.

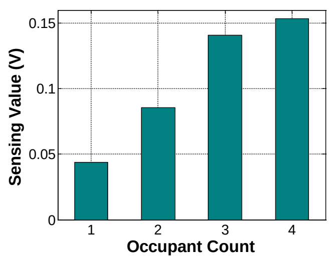
Fig. 6. Impact of ambient illuminance and chip number on sensitivity.

3) Occupant Position and Number: In the previous experiments, we put an occupant right below a sensing unit. Now we let the occupant gradually move away from that center point, and record the corresponding sensing values in Fig. 7(a). Interestingly, whereas the values are generally decreasing in distance, it has a peak at 0.5m. This stems from the fact that the perturbation in reflection incurred by the occupant depends on not only the distance but also occupant's cross section with respect to the unit; the tradeoff between these conflicting factors naturally leads to the peak. This is a nice property as it can effectively increase the FoV of one sensing unit.

We also vary the number of occupants that stand within a circle of radius 1m around the unit. Fig. 7(b) shows an almost linear increase in sensing values with the occupant number, though with some saturation at 4 occupants. Normally, we do not expect to have so crowded situations where more than 4 people stand upon a roughly 3m^2 area, so the results demonstrate the ability of a single unit to count the number of occupants within its FoV, and also explain the need for cooperative sensing with multiple units.



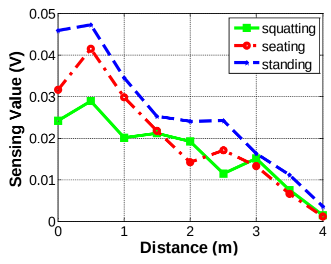
(a) Sensing values vs. distance



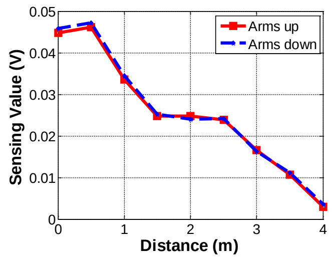
(b) Sensing values vs. occupancy

Fig. 7. Changes in sensing value caused by the occupant position and number.

4) Postures and Gestures of an Occupant: An occupant may have different postures and we consider three typical ones: standing, seating, and squatting (rare). We study the impact of these postures on the sensing values when the occupant is at various distances from a sensing unit. Fig. 8(a) shows that, though squatting does lead to rather low sensing values, standing and sitting cause very close sensing values. This is explained by the fact that the sensing value incurred by the occupant depends on occupant's cross section. So certain occupants may "hide" from CeilingSee by squatting, but this is such a rare posture indoors that it would not cause much trouble to the overall inference performance. An occupant may also change gestures without leaving the monitored area, and such interference should not be responded by CeilingSee. In Fig. 8(b), we show that CeilingSee is virtually insensitive to gesture changes (e.g., waving arms) of an occupant, implying that the occupancy inference function is robust against such an interference. Due to page limit, we omit the demonstrations on robustness against the color and height of an occupant, as well as ambient light variance.



(a) Effect of occupants' postures



(b) Effect of occupants' gesture

Fig. 8. Changes in sensing value caused by different postures and gestures.

III. Occupancy Inference

As illustrated in Fig. 9, the LED sensing readings are first smoothed and sampled to obtain vectors of sensing values (termed *snapshots* hereafter) as described in Section II-B. The snapshots are either directly taken or further differentiated to extract temporal features, and these inputs are fed to a fine-tuned regression module trained for occupancy inference.

A. Spatial Distribution of Sensing Values

When multiple occupants exist in an area monitored by CeilingSee, each sensing unit will produce a sensing value

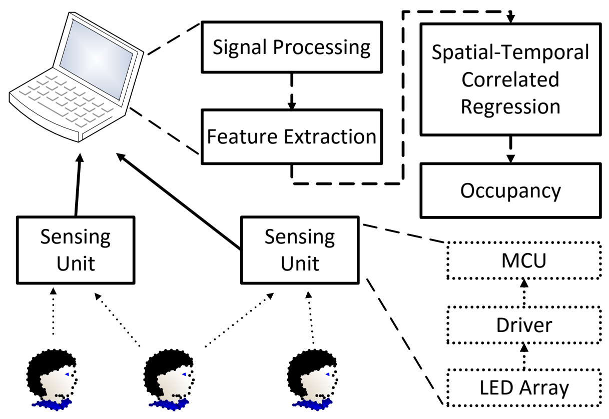


Fig. 9. System architecture. CeilingSee gathers data from all sensing units, pre-processes them to remove noise, and then passes them to the regression module for inferring occupancy.

depending on the spatial distribution of the occupants around it (as studied in Section II-B). We denote the sensing value of a unit by $x_\ell(t)$, where ℓ is the index of the unit and t is the time instant when the value is sampled. Combining all the values produced by the whole system, we end up with a time-varying vector $\mathbf{x}(t) = [x_1(t), x_2(t), \cdots, x_n(t)]$, where n is the number of sensing units. As we always sample $\mathbf{x}(t)$ in a discrete manner, we actually use $\mathbf{x}_i$ to denote a *snapshot* at the i-th time slot, and the spatial distribution of the individual components of a $\mathbf{x}_i$ is correlated with that of the occupants during that time slot. We illustrate a few typical snapshots using our deployment in Fig. 11 of Section IV-A. In Fig. 10, we show 8 snapshots captured when there are 4, 8, 12 occupants and each with two spatial distribution patterns, namely all standing under the sensing units and away from them. A direct observation is that, in general, more occupants usually yield higher overall sensing values and a higher sensing value indicates more occupants around that unit. These observations agree with impact of distance/occupant number on sensing value shown in Fig. 7. More detailed inspections show that, with the same number of occupants at different positions, locating occupants right below the LED sensing units actually leads to lower overall sensing values than moving them away from the units. This stems from our finding in Fig. 7(a). Nevertheless, directly inferring occupancy from individual snapshots through linear regression may yield rather coarse-grained estimations with a low accuracy.

B. Regularized Regression

Given a training set $D = \{\mathbf{x}_i, y_i\}_{i=1,\cdots,m}$ where $\mathbf{x}_i \in \mathbb{R}^n$ denotes the input snapshots, $y_i \in \mathbb{R}$ is the corresponding label (i.e. the occupant count), and m is the cardinality of (sensory) data-label pairs, we need to find a function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ in the form of $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$ to fit the relationship between $\{\mathbf{x}_i\}$'s and $\{y_i\}$'s, where $\mathbf{w} \in \mathbb{R}^n$ is the vector of weight parameters and b is a bias factor (they are to be learned from the training set), and $\langle \cdot, \cdot \rangle$ is the inner product. To avoid the

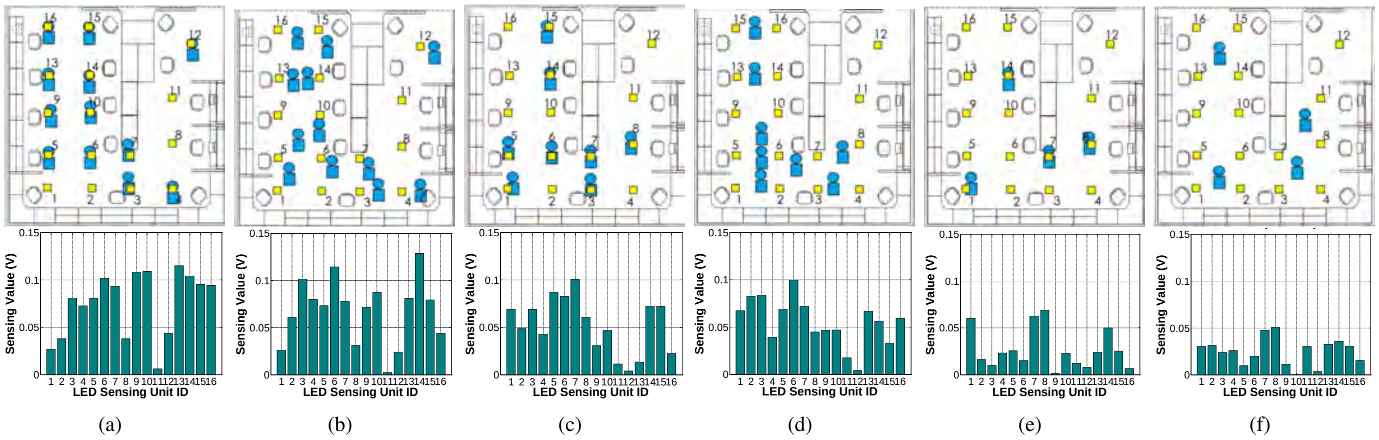


Fig. 10. Snapshot examples: different occupancy patterns lead to different snapshots. Upper figures show the occupancy patterns, with yellow squares represent the sensing units and blue human-shaped marks indicate occupants. Lower figures show the corresponding snapshots. (a) 12 occupants are all under sensing units. (b) 12 occupants are all away from sensing units. (c) 8 occupants are all under sensing units. (d) 8 occupants are all away from sensing units. (e) 4 occupants are all under sensing units. (f) 4 occupants are all away from sensing units.

overfitting issue, we formulate the learning problem of $\mathbf{w}$ as a regularized regression problem by introducing a regularization term on $\mathbf{w}$ as follows,

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left[\sum_{i=1}^{m} \mathcal{L}(f(\mathbf{x}_i), y_i) + \gamma \|\mathbf{w}\|_2^2 \right], \qquad (1)$$

where $\mathcal{L}(\cdot)$ is a loss function and $\gamma \geq 0$ is a tradeoff parameter controlling the relative weight between the loss function and the regularization penalty $\|\mathbf{w}\|_2^2$. Different definitions of the loss function lead to specific regularized regression methods; here we adopt ε-insensitive loss that is described by

$$\mathcal{L}(f(\mathbf{x}), y) = \begin{cases} 0 & \text{if } |f(\mathbf{x}) - y| \leq \varepsilon, \\ |f(\mathbf{x}) - y| - \varepsilon, & \text{otherwise.} \end{cases}$$

This leads to the well-known as Support Vector Regression (SVR) [27]. However, the function $f(\mathbf{x})$ learned by solving (1) can only capture the linear relationship between $\{\mathbf{x}_i\}$'s and $\{y_i\}$'s, whereas the intrinsic relationship between snapshots and the corresponding occupancy counts can be highly nonlinear. To this end, we further formulate the learning problem of $\mathbf{w}$ as a nonlinear regularized regression problem by introducing a nonlinear feature map $\phi(\mathbf{x})$ that maps $\mathbf{x}$ to a Reproducing Kernel Hilbert Space (RKHS),

$$f^*(\mathbf{x}) = \arg\min_{f} \left[\sum_{i=1}^{m} \mathcal{L}(f(\mathbf{x}_i), y_i) + \gamma \|f\|_{\mathcal{H}}^2 \right], \qquad (2)$$

where $\|\cdot\|_{\mathcal{H}}$ is the norm in the RKHS. By using the kernel trick, i.e., $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$, we can obtain the minimizer of the optimization (2):

$$f^*(\mathbf{x}) = \sum_{i=1}^{m} u_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) + b, \qquad (3)$$

where u_i is obtained by solving the dual problem of (2) with the ε-insensitive loss [27]. Our design now focuses on choosing a proper kernel function $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$ and pre-processing $\{\mathbf{x}_i\}$'s so as to improve the inference performance.

C. *Handling Spatial-Temporal Correlations*

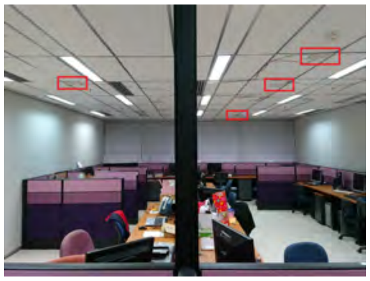
In order to achieve an accurate inference, we fine-tune the regression by taking into account three types of correlations among the training snapshots. As shown in Section III-A, snapshots with similar labels are correlated, while the individual sensing values (collected by sensing units geographically distributed in a monitoring area) have spatial correlations. Moreover, the snapshots taken at consecutive time slots can be correlated depending on whether some occupants move or not.

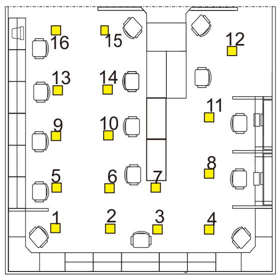
The correlations among snapshots are handled by applying a Gaussian kernel $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\nu \|\mathbf{x}_i - \mathbf{x}_j\|^2}$ with $\nu > 0$. Such a kernel aims to bring down the interferences between two rather "dissimilar" snapshots during the training process. The spatial correlation among sensing values can be handled by pre-processing the training data through the Geographically Weighted Regression (GWR), where each snapshot $\mathbf{x}$ is multiplied by a symmetric matrix W:

$$W(k, \ell) = \begin{cases} e^{-\mu (d_{k\ell}/h)^2} & \text{if} \quad d_{k\ell} < h \\ 0 & \text{otherwise} \end{cases}, \qquad (4)$$

where $d_{k\ell}$ denotes the Euclidean distance between the k-th and ℓ-th sensing units and the "bandwidth" h is set according to the FoV of our sensing units.

The aforementioned approaches may work well when occupant are static. When occupants are moving, the individual snapshots can be rather unstable. Nevertheless, the occupant motion also provide more information, e.g., variance among two consecutive snapshots. Intuitively, minor temporal variances in the snapshots often indicate unchanged occupancy, while major ones may indicate otherwise. To take the advantage of this increased information dimension, we expand every snapshot $\mathbf{x}_i$ by further involving its variance with respect to the previous snapshot, so the new snapshot has the form of $[\mathbf{x}_i, \mathbf{x}_i - \mathbf{x}_{i-1}] \in \mathbb{R}^{2n}$.


(a) Testbed in photo



(b) Sensing units layout

Fig. 11. CeilingSee testbed overview. Sixteen sensing units are deployed on the ceiling covering a 5m×6m indoor area.

D. Incremental Inference

Due to dynamics of a real-world environment, the learning-based system set up using training data pre-collected offline may be out-of-date, and thus perform poorly online. To adapt the system to environmental dynamics, a few training data need to be collected online periodically (normally every week). We also implement an incremental or online algorithm to update the trained SVR model whenever a new training sample is collected in real time [28], which entails a very efficient adaptation to a dynamic environment without the need for retraining from scratch.

IV. System Evaluation

In this section, we evaluate the performance of CeilingSee in terms of inference accuracy, latency and power consumption through extensive field experiments.

A. Experimental Setup

We deploy a CeilingSee testbed in one of our university laboratories; it covers an area of 5m×6m. The testbed consists of 16 LED lighting/sensing units mounted on the ceiling with a height of 2.5m above the floor as shown in Fig. 11(a). We try to deploy the units in 1.25m×1.25m grid, but we have to slightly adjust the positions of some units adapting to the office layout. As briefly presented in Section II-B, each sensing unit includes a 8×12 LED chips array and a re-designed driver. We connect all drivers to a Lenovo ThinkPad laptop computer via serial port, so the laptop acts as a lightweight server processing the sensing data to produce occupancy inferences.

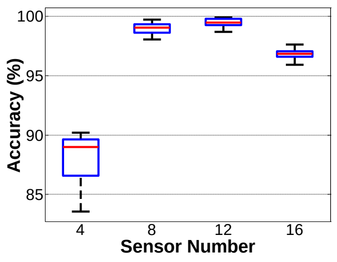
The MCU of our driver, CC2541, is a low-cost power-optimized system-on-chip module, running on up to 32MHz and supporting 12-bit analog-to-digital conversions [29]. Most importantly, CC2541 is designed for Bluetooth Low Energy (BLE). Therefore, although we currently use serial port to connect sensor units to the server, we plan to make CeilingSee a full wireless system by using TI's BLE-STACK [30] to upload sensing data. In our testbed, we run the MCU at 16MHz and let its ADC samples at 500Hz, and the low-pass filter discussed in Section II-B1 is performed by the MCU so that the snapshots are uploaded to the server only at 10Hz, given the low variation rate of the reflection.

This testbed has been running for about 6 months, during which we have kept monitoring the regular occupancy of the deployment site, and we have also invited groups with up to 20 volunteers to perform specific experiments on the system. All the experiments are performed based on two occupancy patterns: i) *static pattern* where occupants stand or sit at arbitrary locations, and 2) *dynamic pattern* where all occupants walk or even run freely in the lab. For each pattern, the number of occupants varies from 1 to 20, and we encourage the occupants to change their postures between standing and sitting, as well as to perform other daily activities during the tests. We gather more than 10,000 snapshots for each pattern; they are all labeled manually.

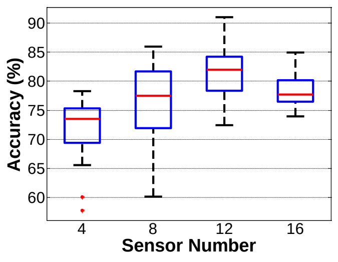
B. Impact of Sensor Density and Training Intensity

Before evaluating the performance of CeilingSee, we firstly experimentally study the two design parameters of CeilingSee, namely how dense the sensing units should be deployed and how much training is needed. For the first aspect, we select 4 (i.e. unit 6, 8, 14 and 12), 8 (i.e. unit 3, 6, 8, 9, 11, 12, 14 and 16), and 12 (i.e. unit 2, 4, 5, 8, 9, 10, 11, 12, 13, 14, 15 and 16) readings out of each snapshot to derive new snapshots, so that we can evaluate the impact of sensor density. For the second aspect, we take a substantial fraction of the labeled data as testing data while using the remaining fraction for the training purpose. We vary the Test to Total Ratio (TTR) from 90% to 99%, where TTR indicates the fraction of data chosen for testing the regression model. To better understand the accuracy performance, we introduce a new accuracy metric that allows for a certain number of miscounts indicated by a non-negative integer τ. In particular, we define *accuracy with miscount tolerance* as $\left(\sum_{i=1}^{c} \mathcal{I}_{|f^*(\mathbf{x}_i)-y_i|\le\tau}\right)/c$, where $\mathcal{I}$ is the indicator function and c is the cardinality..

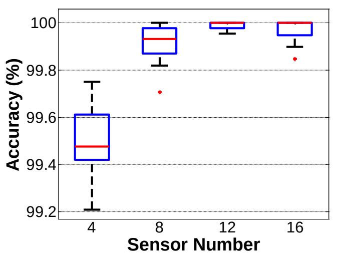
Fig. 12 shows the impact of the number of sensing units. While 8 units appear to already offer very good accuracy for static pattern, 12 units perform the best in both cases and they would be necessary to cope with dynamic situations.



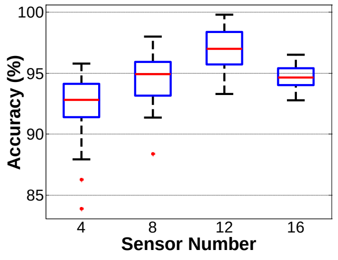
(a) Static pattern, $\tau = 0$.



(b) Dynamic pattern, $\tau = 0$.



(c) Static pattern, $\tau = 1$.



(d) Dynamic pattern, $\tau = 1$.

Fig. 12. Accuracy vs. varying number of sensing units with $\tau = 1$.

Denser deployments beyond 12 units are clearly not beneficial, because it yields a higher dimension of sensed data and complicates the training procedure of the inference model. Fig. 12 further shows the statistics given $\tau = 1$. The relatively low accuracy for dynamic pattern with $\tau = 0$ is mostly due to a single miscount, as raising τ to 1 would allow even 4 units to almost always achieve an accuracy beyond 90% under both patterns, although the better performance of static pattern and the superiority of 12 units still remain. We will continue observing the performance difference between these two patterns, which shall be further explained in Section IV-D. To get the highest accuracy, we will keep using 12 units for the remaining experiments.

Fig. 13 evaluates the impact of training intensity by varying TTR from 90% to 99%. As expected, the higher the TTR (hence less training), the worse the performance (in terms of the mean and variance) is, but the performance degrades in a rather graceful manner. In fact, if one miscount can be tolerated, only 2% training data would be needed for static pattern and 5% for dynamic one. Given our abundant training data, we stick to 90% TTR for the remaining experiments to obtain the best performance.

C. Breakdown of Inference Accuracy

We now study the occupancy inference accuracy with respect to varying occupancy counts; the statistics are reported for both static and dynamic patterns in Fig. 14. We observe that the accuracy based on static pattern is always higher than 97% for all occupancy counts, and tolerating one miscount brings almost all of them to 100%. Though the accuracy for dynamic pattern appears to be rather disappointing, the majority of the miscounting cases involve only one miscount, because setting $\tau = 1$ causes drastic improvements to all occupancy counts.

Under static pattern, the inference accuracy is relatively stable with various occupancy counts, while it is generally degrading as the occupancy count increases for both $\tau = 0$ and $\tau = 1$ under dynamic pattern. This can be largely attributed to the drastic increase in transient states when more occupants are constantly moving, as we shall elaborate in Section IV-D. All in all, if we simply allow one miscount, the inference accuracy based on both static and dynamic pattern can be maintained above 90% for all tested occupancy counts. Therefore, CeilingSee achieves a very promising inference accuracy by using only existing lighting infrastructure. We refrain from comparing CeilingSee with existing proposals at the system level, because, on one hand, it is unfair as they are based on very different technologies, and on the other hand, CeilingSee is not meant to replace other systems, but rather acts as a lightweight complementary solution.

D. Responsiveness and Real Life Scenarios

As CeilingSee may serve as input to indoor energy management systems (e.g., for HVAC), its responsiveness is a concern and hence entails the need for evaluating how quickly CeilingSee can respond to changes in occupancy count. CeilingSee's sensing units configure their sample rates of ADC as 500Hz. A moving average filter in the MCU processes

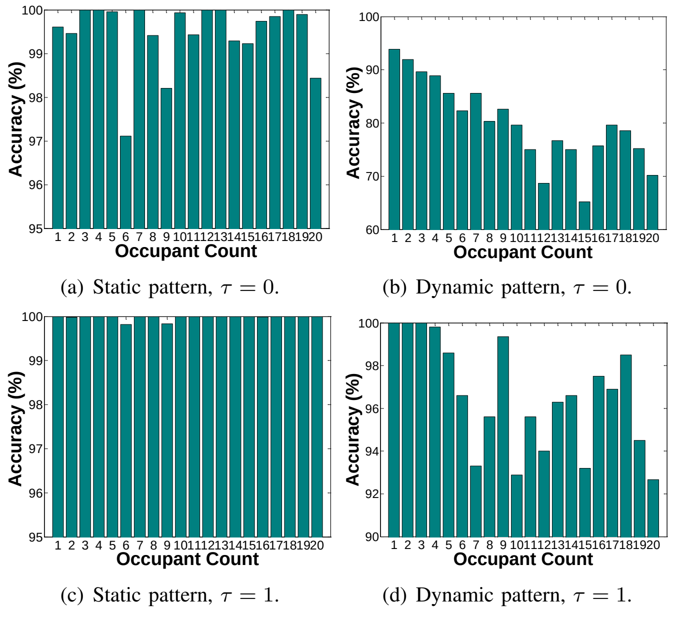


Fig. 14. Accuracy with varying number of occupants.

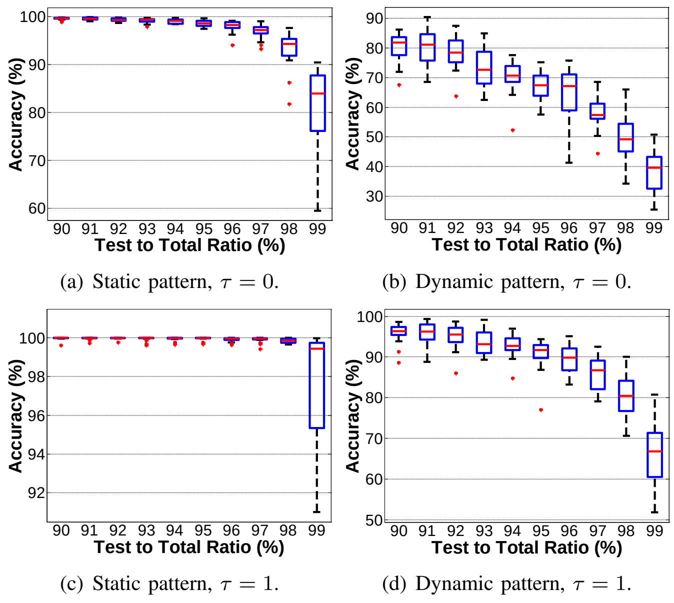


Fig. 13. Accuracy with varying TTR given $\tau = 0$ and $\tau = 1$.

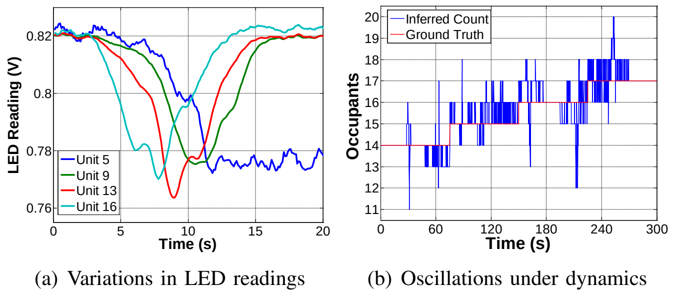
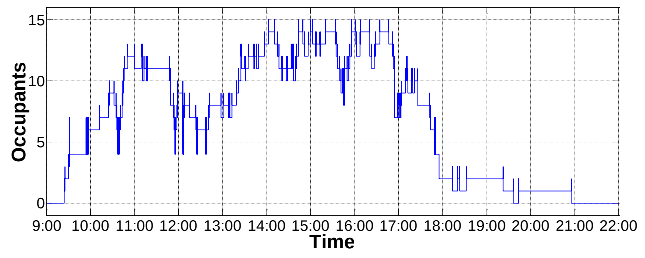
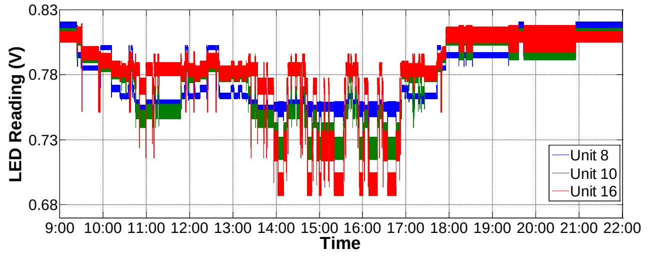


Fig. 15. Dynamic response of CeilingSee.



(a) Occupancy count monitored through a day.



(b) Selected LED readings during the same period.

Fig. 16. A real life monitoring scenario for one day.

the output of ADC and produces snapshots at 10Hz, which causes a 100ms delay. The transmissions of snapshots to the CeilingSee's server via serial port are done at a baud rate of 115,200, consuming less than 1ms. The server takes less than 1ms on average to figure out the occupancy counts by operating our regression algorithm. Therefore, inference latency is about 100ms in total which is mainly consumed by sampling and preliminary data processing on MCU. We could further reduce this delay by running the filter with a smaller window size, but it does not appear to be necessary, because the variations in LED readings take a much slower pace (in seconds), as shown by Fig. 15(a) when an occupant passes by 4 units and stops at the last. In fact, the current high level of responsiveness is one of the main reasons that cause miscounts under dynamic pattern. As shown in Fig. 15(b), the occupancy counts reported by CeilingSee may oscillate between two actual count changes if occupants keep moving. This causes miscounts as shown earlier, but it is the price the system has to pay for instantaneously detecting count changes.

In order to put the performance of CeilingSee into a practical perspective, we choose to report a whole day monitoring data (for 18 February, 2016) among several-month operation of the system. In Fig. 16(a), we plot the inferred occupancy count (without ground truth) from 9am to 22pm. The reported data exhibit a rather plausible pattern: occupants start to arrive in the early morning, they leave for lunch during the noon but come back after lunch; more occupants show up in the afternoon, but most of them leave around 6pm, leaving only a few working till late evening. Between two occupancy count changes, the actual daily activity pattern is a mixture of both static and dynamic patterns: it can be rather stable while most of the occupants are sitting (presumably working), while it fluctuates from time to time due to the activities such as taking a short break, mingling with other colleagues, and so on.

We also arbitrarily choose three sensing units, unit 8, 10, 16, to plot their readings in Fig. 16(b). The trends of these readings are consistent with that of occupancy count. In particular, sensor readings vary only slightly when there are less occupants in the morning, lunch time and off work time, whereas it changes more actively when there are relatively more occupants in the afternoon (as the chance of they being moving becomes higher). If we further look into these readings, for example those around 14:00 and 15:00, fluctuations are so intense that the sampled snapshots can vary rapidly. The resulting large variations in snapshots reduce the effectiveness of regression and hence the inference accuracy. This explains why CeilingSee performs less accurately under dynamic pattern as reported in Section IV-B and IV-C. Fortunately, as we have shown, the miscounts caused by such fluctuations are largely negligible. We could apply Hidden Markov Models (HMMs) [31] to maintain the temporal consistency of occupancy count so as to improve the inference accuracy under dynamic pattern. Nevertheless, it would certainly retard the response of CeilingSee to count actual changes.

E. Energy Consumption

Since CeilingSee utilizes existing lighting infrastructure as light sources, we do not count in the energy consumption for illumination. Consequently, the energy consumption of CeilingSee mainly involves the energy consumed by the driver circuits and microcontrollers of the LED sensing units. The driver circuit of one sensing unit works at a DC voltage of $\pm$ 5V. It consumes 10.1mW when it works in the sensing state, and at most 50mW when working in illuminating state. The energy consumption of the MCU is 24mW. Therefore, the total energy consumption is 34.1mW for a typical CeilingSee sensing unit under its sensing state. As MCUs and drivers are default components in general LED illumination devices, the extra power consumption of our CeilingSee sensing unit is only 10.1mW. This extra power consumption is incurred by the amplifier circuits on re-designed driver. Moreover, since data transmission of sensed data can also be integrated into VLC or Power Line Communication (PLC) [32], the power consumption for data transmission can be neglected. The energy consumption caused by computations on the laptop can also be ignored given that the laptop is working on many other tasks at the same time. In summary, CeilingSee is energy efficient for occupancy inference. We list the power consumptions in Table III.

TABLE III
ENERGY CONSUMPTION OF ONE SENSING UNIT.

	Sensing	Sensing & illuminating
Driver	10.1mW	50.0mW
MCU	24.0mW	24.1mW
Total	34.1mW	74.1mW

V. CONCLUSIONS

In this paper, to efficiently estimate indoor occupancy, we have developed a device-free system, CeilingSee, that piggybacks on existing LED lighting infrastructure. The system consists of two main components: 1) a re-designed LED driver, which leverages LED's photoelectric effect to transform a light emitter to a light sensor, so as to obtain variances in ambient reflection in the form of snapshots in any time, and 2) a machine-learning-based algorithm to infer indoor occupancy using the snapshots as input in real time. To verify the efficiency and effectiveness of the developed system, we have conducted extensive experiments in a testbed covering a 30m^2 laboratory area. The experiment results have shown very promising performance of CeilingSee and hence demonstrated its great potentials to be applied to many smart building applications. In our future work, we plan to further improve the performance of CeilingSee, to develop various intelligent systems that provide personalized services or security monitoring on top of this system, as well as to extend its application to outdoor environments.

References

[1] P. Bahl and V. Padmanabhan, "RADAR: an In-building RF-based User Location and Tracking System," in *Proc. of 19th IEEE INFOCOM*, 2000, pp. 775–784.

[2] Y. Kuo, P. Pannuto, K. Hsiao, and P. Dutta, "Luxapose: Indoor Positioning with Mobile Phones and Visible Light," in *Proc. of 20th ACM MobiCom*, 2014, pp. 447–458.

[3] C. Zhang, F. Li, J. Luo, and Y. He, "iLocScan: Harnessing Multipath for Simultaneous Indoor Source Localization and Space Scanning," in *Proc. of the 12th ACM SenSys*, 2014, pp. 91–104.

[4] C. Zhang, K. Subbu, J. Luo, and J. Wu, "GROPING: Geomagnetism and cROwdsensing Powered Indoor NaviGation," *IEEE Trans. on Mobile Computing*, vol. 14, no. 2, pp. 387–400, 2015.

[5] L. Audin, *Occupancy Sensors: Promise and Pitfalls*, ser. Tech Update, Tu-93-8. Boulder, CO: E Source, Inc., 1999.

[6] R. Dodier, G. Henze, D. Tiller, and X. Guo, "Building Occupancy Detection Through Sensor Belief Networks," *Elsevier Energy and Building*, vol. 38, no. 9, pp. 1033–1043, 2006.

[7] A. Beltran, V. L. Erickson, and A. E. Cerpa, "ThermoSense: Occupancy Thermal Based Sensing for HVAC Control," in *Proc. of the 5th ACM BuildSys*, 2013, pp. 11:1–11:8.

[8] V. L. Erickson, S. Achleitner, and A. E. Cerpa, "POEM: Power-Efficient Occupancy-based Energy Management System," in *Proc. of the 12th ACM/IEEE IPSN*, 2013, pp. 203–216.

[9] S. P. Tarzia, R. P. Dick, P. A. Dinda, and G. Memik, "Sonar-based Measurement of User Presence and Attention," in *Proc. of the 11th ACM UbiComp 2009*, 2009, pp. 89–92.

[10] T. W. Hnat, E. Griffiths, R. Dawson, and K. Whitehouse, "Doorjamb: Unobtrusive Room-level Tracking of People in Homes Using Doorway Sensors," in *Proc. of the 10th ACM SenSys*, 2012, pp. 309–322.

[11] O. Shih and A. Rowe, "Occupancy Estimation Using Ultrasonic Chirps," in *Proc. of the 6th ACM/IEEE ICCPS*, 2015, pp. 149–158.

[12] D. Yang, H. Gonzalez-Banos, and L. Guibas, "Counting People in Crowds with a Real-Time Network of Simple Image Sensors," in *Proc. of the 9th IEEE ICCV*, 2003, pp. 122–129.

[13] V. Erickson, M. Carreira-Perpinan, and A. Cerpa, "OBSERVE: Occupancy-Based System for Efficient Reduction of HVAC Energy," in *Proc. of the 10th ACM/IEEE IPSN*, 2011, pp. 258–269.

[14] C. L. Chen, S. Gong, and T. Xiang, "From Semi-Supervised to Transfer Counting of Crowds," in *Proc. of the 14th IEEE ICCV*, 2013, pp. 2256–2263.

[15] B. Balaji, J. Xu, A. Nwokafor, R. Gupta, and Y. Agarwal, "Sentinel: Occupancy Based HVAC Actuation Using Existing WiFi Infrastructure Within Commercial Buildings," in *Proc. of the 11th ACM SenSys*, 2013, pp. 17:1–17:14.

[16] L. Yang, K. Ting, and M. Srivastava, "Inferring Occupancy from Opportunistically Available Sensor Data," in *Proc. of the 12th IEEE PerCom*, 2014, pp. 60–68.

[17] M. Ibrahim, V. Nguyen, S. Rupavatharam, M. Jawahar, M. Gruteser, and R. Howard, "Visible Light Based Activity Sensing Using Ceiling Photosensors," in *Proc. of the 3rd ACM VLCS Workshop*, 2016, pp. 43–48.

[18] T. Komine and M. Nakagawa, "Fundamental Analysis for Visible-Light Communication System using LED Lights," *IEEE Trans. on Consumer Electronics*, vol. 50, no. 1, pp. 100–107, 2004.

[19] S. Schmid, G. Corbellini, S. Mangold, and T. R. Gross, "LED-to-LED Visible Light Communication Networks," in *Proc. of the 14th ACM MobiHoc*, 2013, pp. 1–10.

[20] H.-Y. Lee, H.-M. Lin, Y.-L. Wei, H.-I. Wu, H.-M. Tsai, and K. C.-J. Lin, "RollingLight: Enabling Line-of-Sight Light-to-Camera Communications," in *Proc. of the 13th ACM MobiSys*, 2015, pp. 167–180.

[21] J. Hao, Y. Yang, and J. Luo, "CeilingCast: Energy Efficient and Location-Bound Broadcast Through LED-Camera Communication," in *Proc. of the 35th IEEE INFOCOM*, 2016, pp. 1629–1637.

[22] T. Li, C. An, Z. Tian, A. T. Campbell, and X. Zhou, "Human Sensing Using Visible Light Communication," in *Proc. of the 21st ACM MobiCom*, 2015, pp. 331–344.

[23] P. Dietz, W. Yerazunis, and D. Leigh, "Very Low-Cost Sensing and Communication Using Bidirectional LEDs," in *Proc. of the 5th ACM UbiComp 2003*, 2003, pp. 175–191.

[24] F. M. Mims, *Siliconnections: Coming of Age in the Electronic Era*. New York: McGraw-Hill, 1986.

[25] LUXEON XF-3535L, "http://www.lumileds.com/products/matrix-platform/luxeon-xf-3535l."

[26] Illuminance, "http://www.engineeringtoolbox.com/light-level-rooms-d_708.html."

[27] A. J. Smola and B. Schölkopf, "A Tutorial on Support Vector Regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[28] J. Ma, J. Theiler, and S. Perkins, "Accurate On-line Support Vector Regression," *Neural Computation*, vol. 15, no. 11, pp. 2683–2703, 2003.

[29] CC2541, "http://www.ti.com/lit/ds/symlink/cc2541.pdf."

[30] BLE-STACK, "http://www.ti.com/product/CC2541/toolssoftware."

[31] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[32] H. Ma, L. Lampe, and S. Hranilovic, "Integration of Indoor Visible Light and Power Line Communication Systems," in *Proc. of 17th IEEE ISPLC*, 2013, pp. 291–296.